5-9-2013

Multistatic Tracking with the Maximum Likelihood Probabilistic Multi-Hypothesis Tracker

Steven C. Schoenecker
schoenecker@comcast.net

Multistatic Tracking with the Maximum Likelihood Probabilistic Multi-Hypothesis Tracker

Steven Schoenecker, Ph.D.

University of Connecticut, 2013

Multistatic sonar tracking is a difficult proposition. The ocean environment typically features very complex propagation conditions, causing low target probabilities of detection and high clutter levels. Additionally, most sonar targets are relatively low speed, which makes it difficult to use Doppler (if available) to separate target returns from clutter returns. The Maximum Likelihood Probabilistic Data Association Tracker (ML-PDA) and the Maximum Likelihood Probabilistic Multi-Hypothesis Tracker (ML-PMHT) — a similar algorithm to ML-PDA — can be implemented as effective multistatic trackers. This dissertation will develop a tracking framework for these algorithms. This framework will focus mainly on ML-PMHT, which has an inherent advantage in that its log-likelihood ratio (LLR) has a simple multitarget formulation, which allows it to be implemented

as a true multitarget tracker. First, this multitarget LLR will be implemented for ML-PMHT, which will give it superior performance over ML-PDA for instances where multiple targets are closely spaced with similar motion dynamics. Next, the performance of ML-PMHT will be compared when it is applied in Cartesian measurement space and in delay-bearing measurement space, where the measurement covariance is more accurately represented. Following this, a maneuver-model parameterization will be introduced that will allow ML-PDA and ML-PMHT to follow sharply maneuvering targets; their previous straight-line parameterization only allowed them to follow moderately maneuvering targets. Finally, a novel method of determining a tracking threshold for ML-PMHT will be developed by applying extreme value theory to the probabilistic properties of the clutter. This will also be done with target measurements, which will allow the issue of trackability for ML-PMHT to be explored. Probabilistic expressions for the maximum values of the LLR surface caused by both clutter and the target will be developed, which will allow for the determination of target trackability in any given scenario.

Multistatic Tracking with the Maximum Likelihood Probabilistic Multi-Hypothesis Tracker

Steven Schoenecker

A.B. Physics, Princeton University, Princeton, NJ, 1995

M.S. Computer Science, Rensselaer at Hartford, Hartford, CT, 2002

A Dissertation

Submitted in Partial Fullfilment of the

Requirements for the Degree of

Doctor of Philosophy

at the

University of Connecticut

2013

APPROVAL PAGE

Doctor of Philosophy Dissertation

Multistatic Tracking with the Maximum Likelihood Probabilistic Multi-Hypothesis Tracker

Presented by

Steven Schoenecker, M.S. Comp. Sci., A.B. Phys.

Major Advisor

Prof. Peter Willett

Co-Major Advisor

Prof. Yaakov Bar-Shalom

Associate Advisor

Dr. Tod Luginbuhl

University of Connecticut

2013

To Lee Ann, Sarah, Peter, and Bethany

ACKNOWLEDGEMENTS

I would like to thank my advisors, Dr. Peter Willett, Dr. Yaakov Bar-Shalom, and Dr. Tod Luginbuhl. It has truly been a privilege working with them, and I have learned a tremendous amount. I would also like to thank the Naval Undersea Warfare Center for supporting me in this effort — I know of very few other places that give their people so much opportunity and support to pursue advanced degrees. It will be a well-made investment. Specifically, I would like to thank many people at NUWC for their guidance, help, support and encouragement. These include my supervisor, Dr. Holly Johnson, Dr. Tod Luginbuhl (one of my advisors), Dr. Tony Ruffa, Jamie McGee, Steve Greineder, Matt Tattersall, Dr. Chris Hempel, Dr. Joe DiBiase, Dr. Philip Ainsleigh, and Dr. Ahmed Zaki. Finally, Dr. Angelo Giannopoulos has provided great support, guidance and mentoring.

Last, but far from least, I owe a tremendous debt of gratitude to my family. To my wife Lee Ann, I cannot thank you enough for all that you have done to support me and to keep the family running the last eight years while I was typing away. To Sarah, Peter, and Bethany, you are the joy of my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

The ultimate goal of a multistatic tracker is to simultaneously estimate the states of an unknown number of targets that produce noisy measurements with a probability of detection (P_d) of less than one — sometimes much less than one — in the presence of clutter. This can be a difficult proposition. Targets parameters are often unknown, and the ocean environment features extremely complex propagation conditions. This can cause high levels of clutter, corrupt target measurements with noise, and make the prediction and use of any feature data, such as received target amplitude, difficult. Finally, most targets are relatively slow-speed, which makes it hard to contrast target measurements from clutter measurements with the use of Doppler. Continuous effort is being made to improve tracking ability in this area.

Up to this point, various multitarget trackers (MTT) have already been applied to the multistatic tracking problem. The more well known algorithms in this group include the Joint Probabilistic Data Association Filter (JPDAF) [33], [8]

the Multiple Frame Assignment Tracker (MFA) [52], [53], the Multiple-Hypothesis Tracker (MHT), both in a hypothesis-oriented (HO) form [55] and a track-oriented (TO) form [42], the Probabilistic Multi-Hypothesis Tracker (PMHT) [7], [68], [69], [70], particle filters [6], the Probability Hypothesis Density (PHD) filter and its variants [44], [45], [31], and the Intensity Filter [66], [67].

This dissertation adds to the MTT body of work by expanding and making improvements to the Maximum Likelihood Probabilistic Data Association (ML-PDA) tracker, and especially (for reasons discussed shortly), the Maximum Likelihood Probabilistic Multi-Hypothesis Tracker. As opposed to all the trackers above, ML-PDA and ML-PMHT are deterministic (non-Bayesian) trackers — they assume a target or targets have some motion that can be parameterized. With this parameterization, along with some assumptions about the targets and the environment, a log-likelihood ratio (LLR) is formulated. The state that optimizes this LLR is then found; if the optimal value of the LLR is greater than a certain threshold, a track is declared; if the optimal value is less than the threshold, it is ignored.

Work on the ML-PDA and ML-PMHT trackers has taken place over the past 20 years, with the majority of development occurring in the last decade. The ML-PDA concept was initially developed in [39]. In [41], the amplitude feature was incorporated into the ML-PDA LLR to improve tracking performance. Next, [23]

used ML-PDA to track a low-observable (LO) electro-optical target; this work also incorporated an adaptive-length sliding batch to the tracker. Subsequently [76], [15], [16] and [20] applied ML-PDA in a multistatic active application, which is how it is currently employed. After this, [78] and [77] began initial implementations of ML-PMHT on multistatic data. Finally, [18] developed a method for determining the ML-PDA tracking threshold, and [17] and [19] expressed and tested the multitarget ML-PDA LLR for two targets.

This dissertation will expand on the above work in several ways. First, it will compare the performance of ML-PDA and an improved version of ML-PMHT. Work in almost all of the above references (with the exception of [78] and [77]) focused on ML-PDA; the majority of work in this dissertation focuses on ML-PMHT for several reasons to be discussed shortly. The main difference between ML-PDA and ML-PMHT lies in the target-measurement assignment model. ML-PDA assumes that at most one measurement can originate from the target in a given frame/scan (throughout this work, these terms will be used interchangeably). ML-PMHT relaxes this requirement and allows any number of measurements to originate from the target in a given scan. While this assumption may reduce the appeal of ML-PMHT to some, it does allow for several very useful results. First, this assumption allows the ML-PMHT LLR to be written in such a way that it can be optimized with the expectation-maximization algorithm [29]. Secondly, and more impor-

tantly, the ML-PMHT LLR can be easily expressed in true multitarget form. It is theoretically possible to write the ML-PDA LLR for multiple targets — this was done for two targets in [17] and [19], but for any more than two or three targets this becomes prohibitively difficult. Additionally, the LLR formulation for ML-PMHT allows for the computation of the Cramér-Rao lower bound (CRLB) in real-time — in [39] and [41] it was shown that the ML-PDA CRLB is of such high dimensionality that it must be computed off-line via Monte Carlo integration. It will be shown that ML-PMHT is a statistically efficient estimator, so its CRLB can be used to represent the uncertainty on the state estimate that the tracker produces.

As a result, ML-PMHT will be implemented as a true multitarget tracker. Its performance will then be compared with the performance of the "legacy" multi-target implementation of ML-PDA. In contrast to ML-PMHT, it is not possible to implement ML-PDA as a true multitarget tracker — it must be implemented as a sequential single-target tracker. It must search the likelihood ratio space to find the strongest target in a batch, assign a track to it, and then excise the measurement in each scan that is most likely associated with this target. Then the search is repeated for the next target, and the target after that, until no more targets are found. These two implementations will be tested with Monte Carlo simulations on six scenarios using a multistatic simulator developed at the Uni-

versity of Connecticut. Part of the multitarget ML-PMHT framework involves calculating the CRLB for the estimator; this work will derive this CRLB, and then it will show that the ML-PMHT estimator on average achieves the CRLB — i.e. the estimator is statistically efficient.

The next topic that will be addressed is the measurement space in which the ML-PMHT tracker operates. Most Kalman-based trackers operate in Cartesian measurement space in order to maintain a linear state-transition relationship; the state-to-measurement relationship often remains non-linear and must use some linearizing approximations. A problem with operating in Cartesian space is that the native measurement space for active sonar is typically time-delay/azimuth. Measurements in this space can be converted to Cartesian measurements following the work of [26], and the measurement covariance can be converted to a Cartesian representation following the work of [25]. However, as the individual measurement errors become large (especially the azimuthal error), this measurement error conversion starts to break down and degrade the performance of the tracker (mainly in terms of in-track percentage and accuracy) — this is sometimes referred to as the "contact lens" problem [73]. Several approaches have been tried to overcome this. In [74] and [12], the authors attempted to represent the measurement covariances in Cartesian space with a Gaussian mixture; alternatively, others have implemented their trackers in the native delay-time/azimuth space using either

an Extended Kalman Filter (EKF) or an Unscented Kalman Filter (UKF) [27]. However, in all these cases, some sort of approximation must be done to get the trackers to function.

In contrast, both ML-PDA and ML-PMHT can function without any linearizing approximations in either Cartesian or delay-azimuth space. ML-PMHT will be implemented in both measurement spaces, and the performance between the two will be measured. This comparison will be again done with Monte Carlo testing; it will use a scenario designed to match the conditions in the Metron dataset [51] that was released in 2009 for use by the Multistatic Tracking Working Group (MSTWG).

The next topic to be addressed will be a maneuver-model parameterization. Up to this point, ML-PDA and ML-PMHT have been implemented as a sliding batch tracker, and all the target motion in a batch was parameterized as straight-line motion. Such straight-line parameterization has been shown to work reasonably well for straight-line targets or for moderately maneuvering targets. However, for targets with more "extreme" maneuvers, tracks tended to become lost; the trackers with their straight-line motion assumptions could not follow the targets through a severe maneuver. In order to address this, a single maneuver will be allowed to occur within a batch; maneuver parameters of maneuver time and maneuver angle are added to state vector (the speed is still kept constant throughout

the batch). This maneuver model will be implemented in two different ways: with a fixed-batch length implementation (MMFL), and a variable-batch length implementation (MMVL) where the front end of the batch is allowed to expand up to twice the original batch size in order to most accurately find the maneuver. Again, this modification will be tested with Monte Carlo trials using the UConn mulitstatic simulator with parameters matched to the conditions in the Metron data set. Lastly, the CRLB development described above will be extended to cover the maneuver model parameterization, and it will be shown that ML-PMHT with this maneuver model parameterization is still statistically efficient.

Finally, this dissertation will explore the topic of trackability for ML-PMHT. To do this, it will develop a representation of the probability density function (PDF) for the maximum point in the ML-PMHT LLR that is caused by clutter, and a representation of the PDF for the maximum point in the LLR that is caused by the target. First, it will be assumed that no target is present, and a single measurement from clutter is considered. The clutter measurement is assumed to be uniformly distributed in measurement space, and the ML-PMHT LLR (again for just a single measurement) is just treated as a random variable transformation. Given the PDF of the clutter measurement, the PDF that "represents" the LLR can be calculated. Next, the PDF of a batch of measurements can be computed through convolution or using characteristic functions. This PDF describes any

point on the ML-PMHT LLR surface. In the actual tracker implementation, the LLR surface is highly non-convex, so it is optimized numerically via methods that will be described in the sequel. Given the PDF of a random point on the LLR surface, this work will "simulate" optimizing the LLR surface by sampling it some number of times. The number of samples N will be the "equivalent" number such that if the samples were ordered, the maximum sample would roughly have the same LLR value that would be achieved by numerical optimization. This problem is equivalent to the order-statistics problem of determining the PDF of the maximum sample from a Gaussian random process. This is done by invoking extreme value theory; the PDF in question will take on a Gumbel distribution [38].

This characterization of the peak PDF in the LLR due to clutter allows for the rapid determination of ML-PMHT tracking thresholds. With this PDF, the optimal detector threshold per the Neyman-Pearson lemma [49] can be set for a desired false-track rate. This will improve and greatly speed up the technique developed for determining the clutter PDF that is presented in [18], which depends on either Monte Carlo simulation or sampling from an actual LLR realization and excising measurements that are associated with any targets.

This same process will then be repeated for the target. As was done with the clutter, the PDF of the maximum value of a "representative" sample set from the

target-measurement created LLR will be calculated. With this, a target's "trackability," to be quantified by a "Tracker Operating Characteristic" curve, can be determined. This curve will be similar to a receiver operating characteristic curve for a binary classifier; it will show in-track percentage vs. false-track probability. First, thresholds that produce various false-track probabilities will be calculated from the clutter PDF. Then, given these thresholds (with their corresponding false-track probabilities) as well as the target distribution, target in-track percentages for each false-track probability can be determined. Each target in-track percentage/false-track probability pair will define a point on the TOC curve; such a curve will show the degree to which a particular target is trackable in a given clutter environment.

This dissertation will be organized as follows: Chapter 2 will provide the background, assumptions used, and log-likelihood ratio development for ML-PDA and ML-PMHT. It will describe how ML-PDA and ML-PMHT are implemented as sliding-batch trackers. Next it will discuss how ML-PDA and early implementations of ML-PMHT were made to track multiple targets, even though their LLRs were only constructed for a single target. After this, the track initiation scheme used for both ML-PDA and ML-PMHT will be described. Chapter 3 will develop a true multitarget tracking framework for ML-PMHT. Using this multitarget implementation, it will provide comparisons between it and the sequential single-

target ML-PDA tracker. In the process, it will develop the CRLB for ML-PMHT and show that ML-PMHT is a statistically efficient estimator. Chapter 4 will compare the performance of ML-PMHT when it is implemented in delay/bearing space versus its performance when it is implemented in Cartesian space. Next it will develop the maneuver-model parameterization for ML-PMHT, and compare the performance of this parameterization with the legacy straight-line parameterization. Chapter 5 will characterize the maximum levels in the ML-PMHT LLR caused by clutter using extreme-value theory. This will allow for simple, straightforward determination of tracking thresholds. In a similar fashion, Chapter 6 will characterize the maximum levels in the ML-PMHT LLR caused by a target. The work from these two chapters will then allow for the determination of trackability for a target with a series of given parameters in a known clutter density. Finally, Chapter 7 will provide conclusions.

For convenience, acronyms used throughout this dissertation are defined in Table 1.1.

Table 1.1: List of acronyms

ML-PMHT	Maximum Likelihood Probabilistic Multi-Hypothesis Tracker
ML-PDA	Maximum Likelihood Probabilistic Data Association
SNR	Signal-to-Noise Ratio
CRLB	Cramér-Rao Lower Bound
FIM	Fisher Information Matrix
LR	Likelihood Ratio
LLR	Log-Likelihood Ratio
GLRT	Generalized Likelihood Ratio Test
MMFL	Maneuvering Model Fixed Length
MMVL	Maneuvering Model Variable Length
NEES	Normalized Estimation Error Squared
EV	Extreme Value
ROC	Receiver Operating Characteristic
TOC	Tracker Operating Characteristic
RV	Random Variable
PDF	Probability Density Function
CDF	Cumulative Distribution Function
PMF	Probability Mass Function
IID	Independent Identically Distributed
CLT	Central Limit Theorem

Finally, the following is a list of journal papers and conference papers that have been published, are submitted, or will be submitted as a result of this research.

Journal Publications:

- S. Schoenecker, P. Willett, Y. Bar-Shalom, and T. Luginbuhl. "Extreme-Value Analysis for ML-PMHT, Part 2: Probability of Target Track Detection and the Fundamental Trackability of Targets," Submitted to *IEEE Transactions on Aerospace and Electronic Systems*.

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "Extreme-Value Analysis for ML-PMHT, Part 1: Threshold Determination for False Track Probability," Submitted to *IEEE Transactions on Aerospace and Electronic Systems*.

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "ML-PDA and ML-PMHT: Comparing Multistatic Sonar Trackers for VLO Targets Using a New Multi-target Implementation," To appear in *IEEE Journal of Oceanic Engineering*, 2013.

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "The ML-PMHT Multi-static Tracker for Sharply Maneuvering Targets," To appear in *IEEE Transactions on Aerospace and Electronic Systems*, 2013.

Conference Publications:

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "ML-PMHT Threshold Determination of False Track Probability Using Extreme-Value Analysis," Submitted to 5^{th} *IEEE Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, St. Martin, December 2013.

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "Comparing a New Multi-target Version of ML-PMHT with ML-PDA for VLO Targets," Submitted to *16th International Conference on Information Fusion*, Istanbul, Turkey, July 2013.

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "Maximum Likelihood Probabilistic Data Association (ML-PDA) Tracker Implemented in Delay-Bearing Space Applied to Multistatic Sonar Data Sets," *Proc. of the SPIE Conference on Signal and Data Processing of Small Targets*, Baltimore, MD, April 2012.

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "A Comparison of the ML-PDA and the ML-PMHT Algorithms," *Proc. 14th International Conference on Information Fusion*, Chicago, IL, July 2011.

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "Maximum Likelihood Probabilistic Multi-Hypothesis Tracker Applied to Multistatic Sonar Data Sets,"

Proc. SPIE Conference on Signal Processing, Sensor Fusion and Target Recognition, Orlando, FL, April 2011.

- R. Georgescu, S. Schoenecker, and P. Willett, "GM-CPHD and ML-PDA Applied to the Metron Multi-Static Sonar Dataset," *Proc. 13th International Conference on Information Fusion*, Edinburgh, Scotland, July 2010.

- S. Schoenecker, P. Willett, and Y. Bar-Shalom. "Maximum Likelihood Probabilistic Data Association Tracker Applied to Bistatic Sonar Data Sets," *Proc. SPIE Conference on Signal and Data Processing of Small Targets.*, Orlando, FL, April 2010.

- R. Georgescu, S. Schoenecker, and P. Willett. "GM-CPHD and MLPDA Applied to the SEABAR07 and TNO-Blind Multistatic Sonar Data," *Proc. 12th International Conference on Information Fusion*, Seattle, WA, July 2009.

Chapter 2

Background

2.1 ML-PDA and ML-PMHT LLR development

The ML-PDA and the ML-PMHT trackers are conceptually very simple algorithms. Some basic assumptions are made about targets and the environment in which they are present. Using these assumptions, a likelihood ratio is formulated, and then optimized. If the LLR at the optimal point is greater than some threshold, a track is deemed to be present.

This section presents the development of the ML-PDA and the ML-PMHT algorithms. It works through the theory behind them, and then it discusses the actual implementation of these algorithms as multistatic active trackers.

2.1.1 ML-PDA LLR

The ML-PDA concept was initially developed in [39] and [40]. Subsequently, in [76], [77], and [20] it was applied in a multistatic active application, which is how we currently employ it. The assumptions used to develop the ML-PDA LR are

[23], [41]:

- A single target is present in each frame with known detection probability P_d. Detections are independent across frames.
- There are zero or one measurements per frame from the target.
- The kinematics of the target are deterministic. The motion is usually parameterized as a straight line, although any other parameterization is valid.
- False detections (clutter) are uniformly distributed in the search volume and their number is Poisson distributed with known density.
- Amplitudes of target and false detections are Rayleigh distributed. The parameter of each Rayleigh distribution is known (although the SNR may be tracked [23]).
- Target measurements are corrupted with zero-mean Gaussian noise with known variance.
- Measurements at different times, conditioned on the parameterized state, are independent.

Based on these assumptions, the ML-PDA log-likelihood ratio can be constructed [11].

Begin by considering a single scan of data. For a scan, the following is a mutually exhaustive and exclusive set

$$\varepsilon_j(i) \triangleq \begin{cases} \text{Measurement } \mathbf{z}_j(i) \text{ is from target}, j = 1, ..., m_i \\ \text{All measurements are clutter}, j = 0 \end{cases} \tag{2.1.1}$$

where m_i is the number of measurements in the i^{th} scan. By the total probability theorem, it is possible to write

$$p[Z(i)|\mathbf{x}] = \sum_{j=0}^{m_i} p[Z(i), \varepsilon_j(i)|\mathbf{x}] \tag{2.1.2}$$

where $Z(i)$ denotes the set of measurements in the i^{th} scan, and $\mathbf{x}$ is the target state parameter vector. The term on the right side of this equation can be expressed as

$$p[Z(i), \varepsilon_j(i)|\mathbf{x}] = p[Z(i)|\varepsilon_j(i), \mathbf{x}]p[\varepsilon_j(i)] \tag{2.1.3}$$

Combining these two equations results in

$$p[Z(i)|\mathbf{x}] = p[Z(i)|\varepsilon_0(i), \mathbf{x}]p[\varepsilon_0(i)] + \sum_{j=1}^{m_i} p[Z(i)|\varepsilon_j(i), \mathbf{x}]p[\varepsilon_j(i)] \tag{2.1.4}$$

Now, the likelihood for the event $\varepsilon_j(i)$ is

$$p[Z(i)|\varepsilon_j(i), \mathbf{x}] = \begin{cases} \frac{1}{V^{m_i}} \prod_{k=1}^{m_i} p_0^\tau[a_k(i)], & j = 0 \\ \frac{1}{V^{m_i-1}} p[\mathbf{z}_j(i)|\mathbf{x}] p_1^\tau[a_j(i)] \prod_{k=1, k\neq j}^{m_i} p_0^\tau[a_k(i)] & j = 1, ..., m_i \end{cases} \tag{2.1.5}$$

Here, V is the search volume (in measurement space), $p_1^\tau(a)$ is the amplitude likelihood for the target, $p_0^\tau(a)$ is the amplitude likelihood for clutter, and $p[\mathbf{z}_j(i)|\mathbf{x}]$ is a target-centered Gaussian. (The forms of the target and clutter amplitude

likelihoods will be defined in Section 2.1.3.) Define the amplitude likelihood ratio as

$$\rho_j(i) = \frac{p_1^\tau(a)}{p_0^\tau(a)} \tag{2.1.6}$$

Now, equation (2.1.5) can be re-written as

$$p[Z(i)|\varepsilon_j(i), \mathbf{x}] = \begin{cases} \frac{1}{V^{m_i}} \prod_{k=1}^{m_i} p_0^\tau[a_k(i)], & j = 0 \\ \frac{1}{V^{m_i-1}} p[\mathbf{z}_j(i)|\mathbf{x}] \rho_j(i) \prod_{k=1}^{m_i} p_0^\tau[a_k(i)] & j = 1, ..., m_i \end{cases} \tag{2.1.7}$$

At this point, the event probability $\varepsilon_0(i)$ is expressed as

$$p[\varepsilon_0(i)] = (1 - P_d)\mu(m_i) \tag{2.1.8}$$

where P_d is the probability of target detection in a scan, and $\mu(m)$ — the probability that m events occur – is a Poisson PMF. Likewise, the probability for $\varepsilon_j(i)$ — the probability of the event that the j^{th} measurement is from the target and the other $m_i - 1$ measurements are from the clutter is

$$p[\varepsilon_j(i)] = \frac{P_d}{m_i}\mu(m_i - 1) \qquad 1 < j < m_i \tag{2.1.9}$$

Combining (2.1.4), (2.1.7), (2.1.8), and (2.1.9) gives the result

$$\begin{aligned} p_1[Z(i)|\mathbf{x}] =& (1 - P_d)\mu(m_i)\frac{1}{V^{m_i}} \prod_{j=1}^{m_i} p_0^\tau[a_j(i)] + \cdots \\ & \frac{P_d}{m_i}\mu(m_i - 1)\frac{1}{V^{m_i-1}} \prod_{j=1}^{m_i} p_0^\tau[a_j(i)] \sum_{j=1}^{m_i} p[\mathbf{z}_j(i)|\mathbf{x}]\rho_j(i) \end{aligned} \tag{2.1.10}$$

where P_d is the probability of target detection in a scan, and $\mu(m)$ — the probability that m events occur – is a Poisson PMF.

It is desired to have a likelihood ratio, so (2.1.10) is divided by the likelihood of all measurements originating from clutter, which is expressed as

$$p_0[Z(i)|\mathbf{x}] = \frac{1}{V^{m_i}}\mu(m_i)\prod_{j=1}^{m_i} p_0^{\tau}[a_j(i)] \tag{2.1.11}$$

Dividing (2.1.10) by (2.1.11) produces the likelihood ratio for a scan of data

$$\frac{p_1[Z(i)|\mathbf{x}]}{p_0[Z(i)|\mathbf{x}]} = 1 - P_d + \frac{P_d}{\lambda}\sum_{j=1}^{m_i} p[\mathbf{z}_j(i)|\mathbf{x}]\rho_j(i) \tag{2.1.12}$$

Here p_1 is used denote the likelihood expressed by (2.1.10), while p_0 is used to denote the likelihood for all measurements being from clutter (these should not be confused with p_0^{τ} and p_1^{τ}, the amplitude likelihoods), and λ is the spatial clutter density. Now, typically, the ML-PDA algorithm processes a batch of data (i.e. several scans). Recalling from the ML-PDA assumptions that measurements at different times, conditioned on the parameterized state, are independent, the likelihood ratio of a batch of measurements is just the product of the scan likelihood ratios. Finally, as is commonly done, the likelihood ratio (LR) is converted to a log-likelihood ratio (LLR). The final expression that results is

$$\Lambda_0(Z, \mathbf{x}) = \sum_{i=1}^{N_w} \log\left\{1 - P_d + \frac{P_d}{\lambda}\sum_{j=1}^{m_i} p[\mathbf{z}_j(i)|\mathbf{x}]\rho_j(i)\right\} \tag{2.1.13}$$

Here N_w is the number of scans, and Z is the entire set of measurements. Finally, in processing the ML-PDA LLR, a constant is often subtracted out of the expression so the floor of the LLR is zero

$$\Lambda(Z, \mathbf{x}) = \sum_{i=1}^{N_w} \log\left\{1 + \frac{P_d}{\lambda(1 - P_d)}\sum_{j=1}^{m_i} p[\mathbf{z}_j(i)|\mathbf{x}]\rho_j(i)\right\} \tag{2.1.14}$$

2.1.2 ML-PMHT LLR

The Maximum Likelihood Probabilistic Multi-Hypothesis Tracker is very similar to the ML-PDA algorithm. The ML-PMHT likelihood ratio was first formulated for the PMHT algorithm in [7], [68], [69] and [70]. It was implemented with a maximum-likelihood formulation in a bistatic active application in [77] and [78], which is how it is currently employed. The assumptions that go into it are almost exactly the same as those listed above for ML-PDA, with one (major) exception. Instead of allowing only zero or one measurement in a scan to be assigned to the target, ML-PMHT allows any number of measurements in a scan to be assigned to the target. While this would seem to add complexity to the problem, it actually makes the log-likelihood ratio much easier to develop. For a single scan of data, the likelihood is expressed as

$$p_1^\dagger[Z(i)|\mathbf{x}] = \prod_{j=1}^{m_i} \left\{ \frac{\pi_0}{V} p_0^\tau[a_j(i)] + \pi_1 V p[\mathbf{z}_j(i)|\mathbf{x}] p_1^\tau[a_j(i)] \right\} \tag{2.1.15}$$

where π_0 is the prior probability that a measurement is from clutter, π_1 is the prior probability that a measurement is from the target, and V is the search volume. The likelihood for all the measurements being from clutter is

$$p_0^\dagger[Z(i)|\mathbf{x}] = \frac{1}{V^{m_i}} \prod_{j=1}^{m_i} p_0^\tau[a_j(i)] \tag{2.1.16}$$

Taking the ratio of these two equations gives the ML-PMHT LR for a scan of data

$$\frac{p_1^\dagger[Z(i)|\mathbf{x}]}{p_0^\dagger[Z(i)|\mathbf{x}]} = \prod_{j=1}^{m_i} \left\{ \pi_0 + \pi_1 V p[\mathbf{z}_j(i)|\mathbf{x}] \rho_j(i) \right\} \tag{2.1.17}$$

Again, since measurements at different times, conditioned on the state, are independent, the likelihood ratio of a batch is just a product over i of N_w terms on the right-hand side of (2.1.17). Doing this and then taking the logarithm produces the ML-PMHT log-likelihood ratio for a batch:

$$\Lambda_0^\dagger(\mathbf{x}, Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \left\{ \pi_0 + \pi_1 Vp[\mathbf{z}_j(i)|\mathbf{x}]\rho_j(i) \right\} \tag{2.1.18}$$

As with the case of ML-PDA, in the actual ML-PMHT implementation, a constant is subtracted out of the expression so the floor of the LLR is at zero

$$\Lambda^\dagger(\mathbf{x}, Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \left\{ 1 + \frac{\pi_1}{\pi_0} Vp[\mathbf{z}_j(i)|\mathbf{x}]\rho_j(i) \right\} \tag{2.1.19}$$

2.1.3 Amplitude likelihood ratios

Here, we discuss the amplitude likelihood ratios (LRs) that were developed in [41], [9], and [23]. In these works, the amplitudes from both clutter and target returns were assumed to follow a thresholded Rayleigh distribution. This is a result of assuming a target/scatterer has a cross-section that follows a Swerling I fluctuation model [14], [71], [46], [47]. The clutter amplitude distribution is written as

$$p_0^\tau(a) = 2ae^{\tau^2}e^{-a^2} \qquad a \geq \tau \tag{2.1.20}$$

where τ is the detector threshold, in units of amplitude. The target distribution follows another thresholded Rayleigh distribution

$$p_1^\tau(a) = \frac{2a}{\sigma^2} e^{\frac{\tau^2}{\sigma^2}} e^{-\frac{a^2}{\sigma^2}} \qquad a \geq \tau \tag{2.1.21}$$

where σ^2 is the expected intensity (amplitude squared) from the target. The ratio of these gives the expression for the target LR

$$\rho(a) = \frac{1}{\sigma^2} \frac{e^{\tau^2/\sigma^2} e^{-a^2/\sigma^2}}{e^{\tau^2} e^{-a^2}} \qquad a \geq \tau \tag{2.1.22}$$

2.1.4 Tracking implementation

This section describes the implementation of ML-PDA and ML-PMHT as a multistatic multitarget trackers. First we discuss how ML-PDA and ML-PMHT are set up as sliding batch trackers. Next, we present the overall track logic. After this, the optimization used to find the maxima in both the ML-PDA and ML-PMHT LLRs is covered. Finally, we discuss the initialization routine used for global optimization.

In the tracking implementation described below, there is actually very little difference between using ML-PDA and ML-PMHT.[1] The sliding batch, track framework, and track initialization are practically the same for both algorithms; the difference is just the actual LLRs that are implemented and optimized. In the case of ML-PDA the LLR described by (2.1.14) is implemented; in the case of ML-PMHT, the LLR described by (2.1.19) is implemented.

[1] When implemented as sequential single-target trackers. In the sequel, ML-PMHT will be implemented in true multitarget form.

Tracking with a sliding batch

In order to work as a multistatic tracker, both ML-PDA and ML-PMHT must make an assumption about the target motion; as is stated above in Section 2.1.1, one of the assumptions is that the target motion can be parameterized. The most logical motion to parameterize is a straight line, and this is precisely what is done initially for the tracking algorithms. We want the ability to track maneuvering targets, so the assumption is made that "reasonable" target motion can be approximated as a series of straight-line segments. This is done in the actual tracking framework by implementing the data passed to the tracker in the form of a sliding batch. Figure 2.1 illustrates this. For a typical data update/ping repetition rate (60 seconds), a series of scans, typically covering approximately 10 minutes of data, was fed as a batch to the tracker. The track cycle (discussed in the next section) was run, and then the batch was slid forward by 2 minutes of data.

There is an inherent tradeoff in the length of the batch. In terms of being able to track a maneuvering target, the batch should be as short as possible in order to catch as many maneuvers as possible. However, in terms of obtaining "data integration," it is desired to have the batch as as long as possible. The numbers presented here were found empirically to meet this tradeoff nicely.

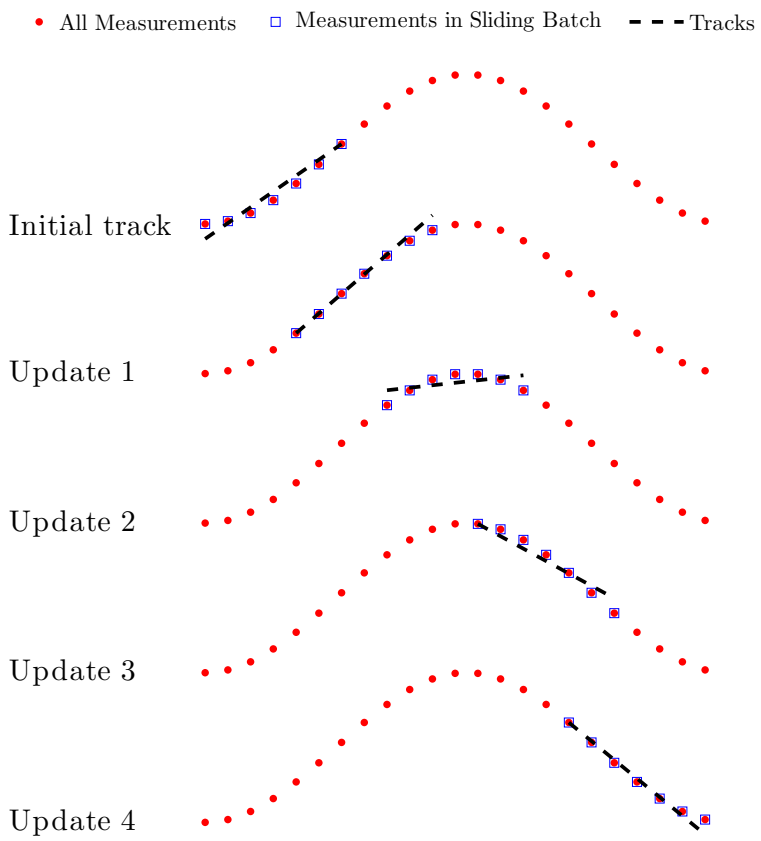

Fig. 2.1: Illustration of sliding-window batch processing

Tracker framework

Using the above sliding batch implementation, ML-PDA and ML-PMHT were incorporated into a tracking framework. The tracking framework can be divided into two basic parts: searching for new tracks and updating existing tracks. At the very start of a tracking process, there are obviously no existing tracks, so an initial batch of data is passed to the track initialization routine (described in Section 2.1.4). The batch is slid forward, and the (now) existing tracks that were discovered during the very first track initialization routine are updated.

The ML-PDA and ML-PMHT LLRs given by (2.1.14) and (2.1.19) are for a single target only; in order to be able to track multiple targets in a scenario, they must be implemented as sequential single-target trackers. (In Section 3.4.2, ML-PMHT will be implemented as a true multitarget tracker.) This is done by cycling through the existing tracks, one at a time. A track is taken from the existing track queue; the track state is described by the track parameter vector

$$\mathbf{x} = (x_0 \;\; \dot{x} \;\; y_0 \;\; \dot{y})^T \tag{2.1.23}$$

where x_0 and y_0 are the Cartesian positions of the target at the beginning of the batch, and $\dot{x}$ and $\dot{y}$ are the Cartesian velocities that describe the motions throughout the batch. Since the motion throughout the batch is assumed to be straight-line, (2.1.23) can be used to describe/calculate the target position at any

time that falls within the batch being processed.[2]

ML-PDA and ML-PMHT are, as their names suggest, maximum likelihood routines. The idea is to find the global maximum of their LLRs; if this global optimum is over a certain threshold, it is declared a target. Otherwise, the peak is deemed to be due to clutter. The LLRs must be optimized numerically (i.e. there is not a closed-form solution to the global maximum of the LLRs); to do this, an initialization point is required. If a valid target is present at time t_1 with a state parameter vector $\mathbf{x}_1$, it is logical to assume that the best place to start looking for a target at the next time update is at $\mathbf{x}_2$, where $\mathbf{x}_2$ is given by

$$\mathbf{x}_2 = \mathbf{F}\mathbf{x}_1; \tag{2.1.24}$$

The state transition matrix $\mathbf{F}$ is

$$\mathbf{F} = \begin{bmatrix} 1 & T & 0 & 0 \\ & & & \\ 0 & 0 & 1 & T \end{bmatrix} \tag{2.1.25}$$

where T is the tracker update interval. At this point, either the ML-PDA or the ML-PMHT LLR is optimized, using $\mathbf{x}_2$ as an initialization point. When a new optimum point is found, the track (with its optimized LLR) is put through a series of checks. First, the track undergoes a kinematic check — if the magnitude of the velocity is greater than some large value, the track is dropped immediately. The

[2] Having the positional components of (2.1.23) represent the front of the batch is not a requirement — they could actually be used to describe *any* fixed reference point in the batch; a point just needs to be selected, and for this application, the front of the batch was picked.

reason for this is that the solution determined by the maximum LLR has given a physically ridiculous answer (for example, no maritime target could possibly have a velocity of 1000 meters/sec). The (high) velocity components affect the positional components of the found state $\mathbf{x}_{max}$ as well; if the velocity components are incorrect by orders of magnitude, the positional components will also be off by orders of magnitude. The state parameter vector for this track is clearly nowhere near its true target, so it is dropped immediately.

After the kinematic "sanity" check, the continued existence of the track is dependent on its track health. Each track is assigned at every update a health value ranging from zero to five in integer units. If the LLR value at the optimized position described by $\mathbf{x}_{max}$ is greater than the tracking threshold, and the track health is less than the maximum value allowed (five), then the track health is incremented by one unit. If the track health is already at the maximum value, it is kept constant. If the LLR value is less than the threshold, then the track health is decreased by one. If this decrease results in a track health value of one or greater, the track is maintained. When the track health drops to zero, the track is dropped.

The idea behind this track health scheme is that overall, tracks that are "weak" should be able to be dropped easily, while tracks that are strong should be allowed some "misses" before they are terminated. Tracks are initiated at a health of two

with the idea that newly found tracks are expected to be relatively weak, but they should be allowed one miss (in terms of not meeting the LLR threshold on the update after initiation) before termination.

Finally, after a successful track update, measurements associated with the track are excised. This is necessary so other tracks do not use these same measurements, or more likely, a new (and duplicate) track is initiated on these measurements. This step is very key in making the algorithms multitarget trackers. For ML-PDA, the measurement in a scan within a certain gate that has the highest probability of being associated with the track is excised. For ML-PMHT, all measurements in a scan with an association probability [described below in (2.1.27)] of $w_j(i) > 0.1$ are excised.

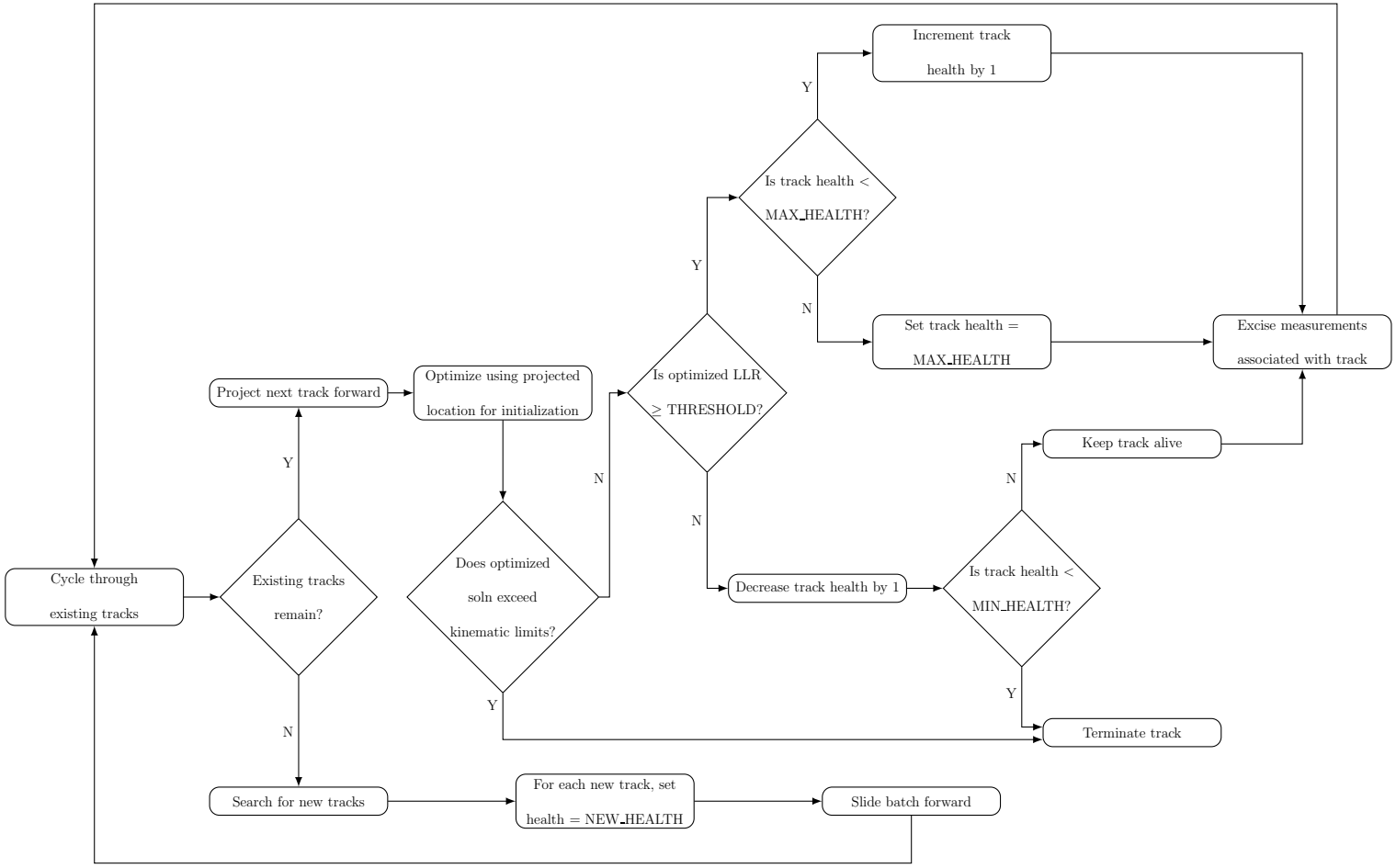

Fig. 2.2: Track management framework

Optimization

The optimization routine for the tracking framework is one of the areas where there can be a difference between (sequential single-target) ML-PDA and ML-PMHT. The ML-PDA algorithm must be optimized numerically. Throughout this research, two types of nonlinear optimization schemes were used for ML-PDA. At first, a customized conjugate-gradient approach was used [65], [54], [50], [13]. However, it was found that the quasi-Newton method present in Matlab's Optimization Toolbox, which implements the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method [21], [32], [37], [64] works both faster and more accurately. In the final tracking framework, this was used for optimizing the ML-PDA algorithm.

An advantage of the ML-PMHT log-likelihood ratio formulation is that it can be iteratively optimized with a closed-form expression using expectation maximization [29]. As long as there is a linear relationship between the state $\mathbf{x}$ and the predicted measurement $\hat{\mathbf{z}}$, the cost function $J(\mathbf{x}, Z)$ can be written as

$$J(\mathbf{x}, Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \Big\{ [\mathbf{z}_j(i) - \mathbf{H}\mathbf{x}]^T \mathbf{R}_{ij}^{-1} \times [\mathbf{z}_j(i) - \mathbf{H}\mathbf{x}] + \ln(|2\pi \mathbf{R}_{ij}|) \Big\} w_j(i) \tag{2.1.26}$$

Here, $\mathbf{H}$ is the measurement matrix, $\mathbf{R}_{ij}$ is the measurement covariance matrix for the j^{th} measurement in the i^{th} scan, and $w_j(i)$ is the association probability of the measurement. The EM algorithm for this case involves iteratively calculating $w_j(i)$ and then using this value to solve for the minimum of equation (2.1.26) [7].

The expression for $w_j(i)$ is

$$w_j(i) = \frac{\frac{\pi_1}{\sqrt{|2\pi\mathbf{R}_{ij}|}} e^{-\frac{1}{2}[\mathbf{z}_j(i)-\hat{\mathbf{z}}]^T \mathbf{R}_{ij}^{-1}[\mathbf{z}_j(i)-\hat{\mathbf{z}}]}}{\frac{\pi_0}{V} + \frac{\pi_1}{\sqrt{|2\pi\mathbf{R}_{ij}|}} e^{-\frac{1}{2}[\mathbf{z}_j(i)-\hat{\mathbf{z}}]^T \mathbf{R}_{ij}^{-1}[\mathbf{z}_j(i)-\hat{\mathbf{z}}]}} \tag{2.1.27}$$

In equation (2.1.26), the relationship between the predicted measurement $\hat{\mathbf{z}}$ and the state is linear

$$\hat{\mathbf{z}} = \mathbf{H}\mathbf{x} \tag{2.1.28}$$

This relationship is valid when the measurements are two-dimensional (typically x and y Cartesian coordinates), and the state vector is given by (2.1.23). The measurement matrix is given by

$$\mathbf{H} = \begin{bmatrix} 1 & t & 0 & 0 \\ 0 & 0 & 1 & t \end{bmatrix} \tag{2.1.29}$$

In this case, t is the time of the measurement, and the initial time $t = 0$ is at the beginning of the batch. When the relationship described by (2.1.28) holds, the minimization of the cost function in (2.1.26) is a simple vector quadratic minimization, the solution to which is easily worked out to be

$$\mathbf{x} = \Big[\sum_{i=1}^{N_w}\sum_{j=1}^{m_i} w_j(i)\mathbf{H}^T\mathbf{R}_{ij}^{-1}\mathbf{H}\Big]^{-1} \times \sum_{i=1}^{N_w}\sum_{j=1}^{m_i} w_j(i)\mathbf{H}^T\mathbf{R}_{ij}^{-1}\mathbf{z}_j(i) \tag{2.1.30}$$

When Doppler is added to the measurement, the relationship between the predicted measurement and the state vector is no longer linear, and the above solution

can not be used. In order to avoid this problem, the new (three-dimensional) predicted measurement can be linearized about some initial state vector $\mathbf{x}_0$. In this case, the linearized measurement is

$$\hat{\mathbf{z}}(\mathbf{x}) \approx \begin{bmatrix} \mathbf{H} \\ \\ \nabla\tilde{r}(\mathbf{x}_0)^T \end{bmatrix}\mathbf{x} + \begin{bmatrix} 0 \\ \\ 0 \\ \\ \tilde{r}(\mathbf{x}_0) - \nabla\tilde{r}(\mathbf{x}_0)^T\mathbf{x}_0 \end{bmatrix} \tag{2.1.31}$$

In this equation, $\tilde{r}$ is the bistatic Doppler, a function of the state vector and the positions and velocities of the source and receiver [26]. Now, the vector on the right-hand side of this equation can be shifted to the left-hand side of the equation to produce a modified predicted measurement

$$\tilde{\mathbf{z}} = \hat{\mathbf{z}} - \begin{bmatrix} 0 \\ \\ 0 \\ \\ \tilde{r}(\mathbf{x}_0) - \nabla\tilde{r}(\mathbf{x}_0)^T\mathbf{x}_0 \end{bmatrix} \tag{2.1.32}$$

and

$$\bar{\mathbf{H}} = \begin{bmatrix} \mathbf{H} \\ \\ \nabla\tilde{r}(\mathbf{x}_0)^T \end{bmatrix} \tag{2.1.33}$$

so we can write

$$\tilde{\mathbf{z}} = \bar{\mathbf{H}}\mathbf{x} \tag{2.1.34}$$

The linear relationship between the (modified) predicted measurement and the state again holds, so equation (2.1.30) can be used for the optimization of the cost function and the solution for the state vector $\mathbf{x}$ for optimizing ML-PMHT.

The EM algorithm was used initially for ML-PMHT optimization. However, it was discovered that the quasi-Newton method from the Matlab Optimization Toolbox (which was used for ML-PDA optimization) works faster and just as well in some cases, and when Doppler is being processed, it actually works better (due to the fact the above approximations are required when using EM to process Doppler), so in this work, this optimization scheme was primarily used. However, when Matlab is not available, the EM scheme remains valid and useful for optimizing ML-PMHT.

Track initialization

Several types of track initialization were tried in the ML-PDA and the ML-PMHT tracking frameworks. These included one-point initialization, two-point initialization [10] and a simple grid initialization. In the end a customized "intelligent" grid initialization was found to work the best. The ultimate goal of any track initialization scheme is to (obviously) find all new targets that present themselves, while minimizing or avoiding altogether any false tracks. Additionally, while it is desired to initialize tracks on all new targets that present themselves as peaks in the LLR surface, it is necessary to avoid creating duplicate tracks — i.e. more than one track for a single target.

The intelligent grid initialization scheme is implemented as follows. First, as is described above, measurements associated with already existing tracks are excised

from the data, since it would be incorrect to initialize a new track on a target that already has a track assigned to it. Next a grid is created over the existing 2-dimensional Cartesian space. Typically, the spacing is set in both the x and y directions at 1000 distance units. At each (x, y) location, a "velocity fan" is created. This velocity fan covers 17 different velocities; the fan is constructed from the initial search directions $\theta \in \{0°,\ 45°,\ 90°,\ 135°,\ 180°,\ 235°,\ 270°,\ 315°\}$. For each direction, speeds of 3 units per second and 6 units per second are used; additionally a speed of 0 (which obviously requires no direction) is included. Such a setup is a good compromise between finely sampling the target parameter space and minimizing the computational requirements that come out of having a very large number of points to sample. Over a 100,000 square-unit grid (a representative size), this spacing gives approximately 10^6 Cartesian points; at each point, 17 speed combinations are tried for a total number of combinations of 1.7×10^7. The track initialization routine is started by calculating the ML-PMHT LLR at each one of these points.[3] The calculated LLRs are ordered, and then the top 200

[3] Either the ML-PDA or the ML-PMHT LLR can be used, and both were tested. In the end, the ML-PMHT LLR was used due to the fact that when the data followed the ML-PDA target assignment model — zero or one target originated measurement present in a scan, there was no appreciable difference in track initialization performance between the two. However, when the data followed the ML-PMHT target-measurement assignment model, ML-PMHT outperformed ML-PDA.

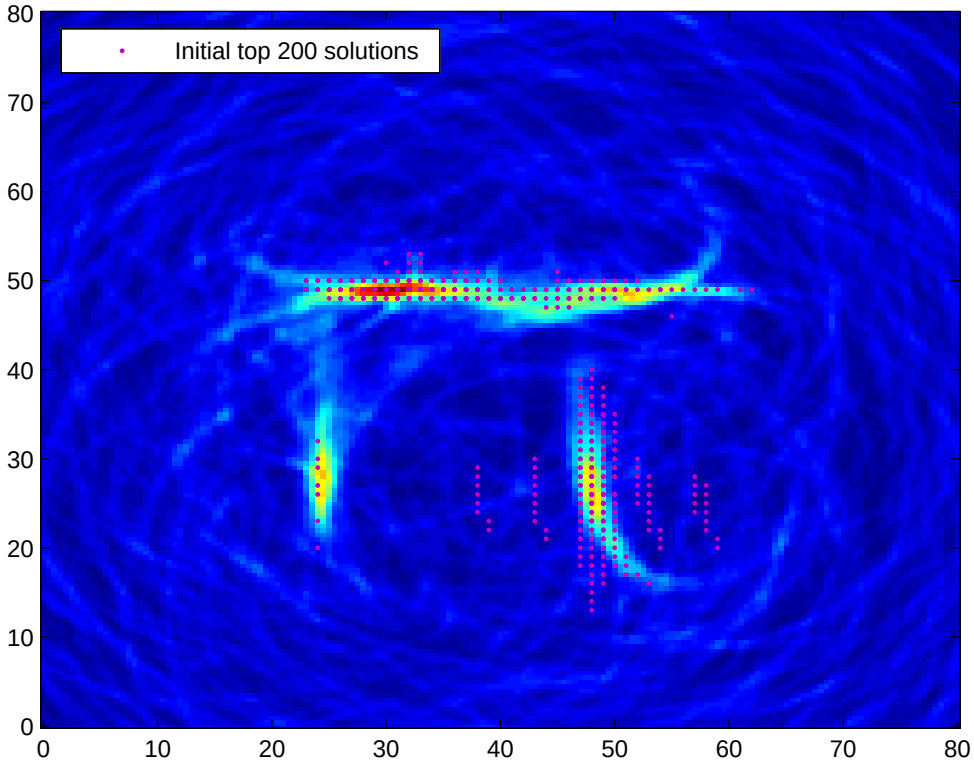

Fig. 2.3: Track initialization example. ML-PMHT LLR 2-D surface shown with top 200 initial solutions.

LLR values are stored (denoted by ς_0), along with the initial target parameters that produce them. An example of this is shown in Figure 2.3 — this plot shows the ML-PMHT LLR surface plotted versus x and y coordinates (the velocities for the plot are fixed at 0 units/sec). Also shown are the top 200 LLR values — note that they overlay the four visual peaks in the LLR surface (in this particular example, there are four targets present). At this point, these initial solutions are "reduced" in order to prevent duplicate targets from being initialized. The

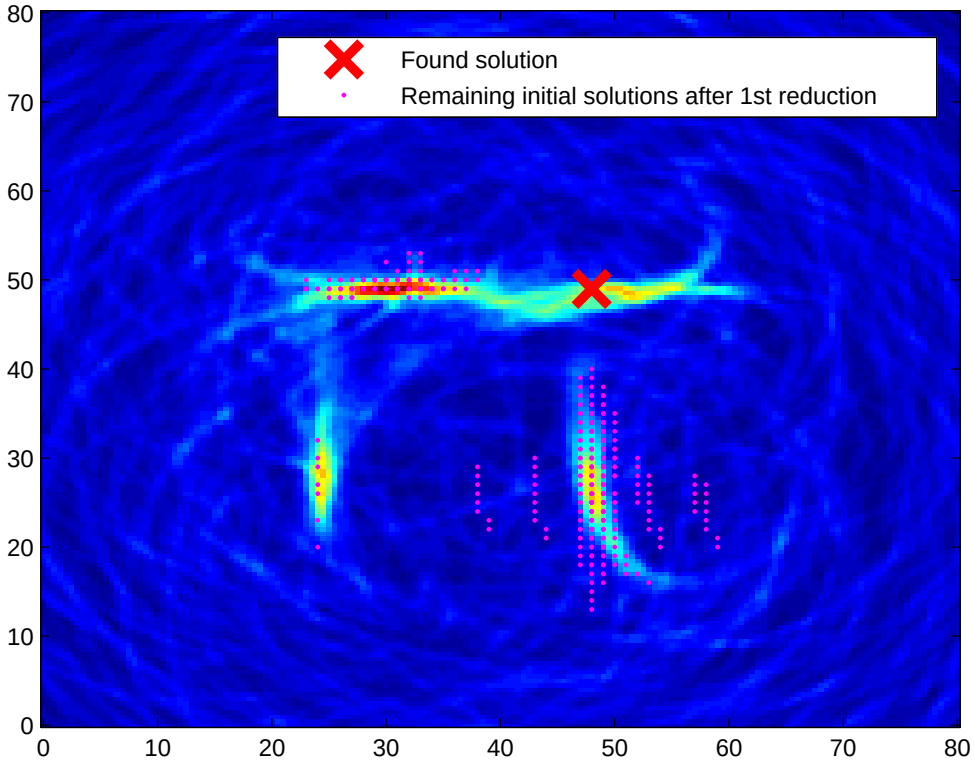

Fig. 2.4: Track initialization example. ML-PMHT LLR 2-D surface shown with first found solution, and initial solutions remaining after first reduction.

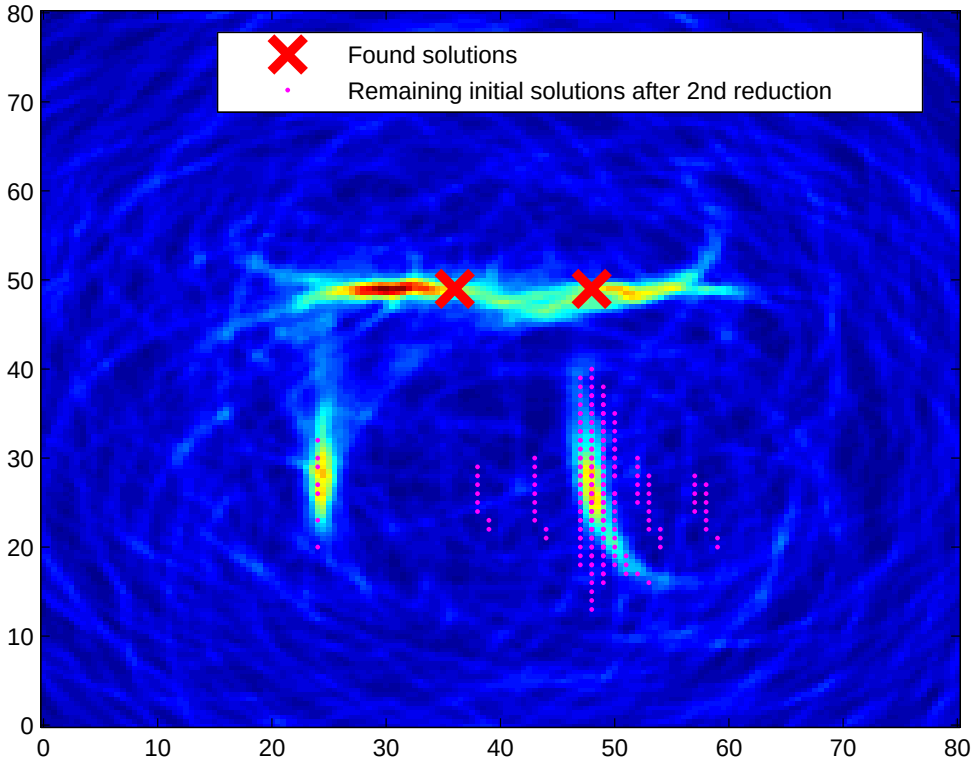

Fig. 2.5: Track initialization example. ML-PMHT LLR 2-D surface shown with first two found solutions, and initial solutions remaining after first two reductions.

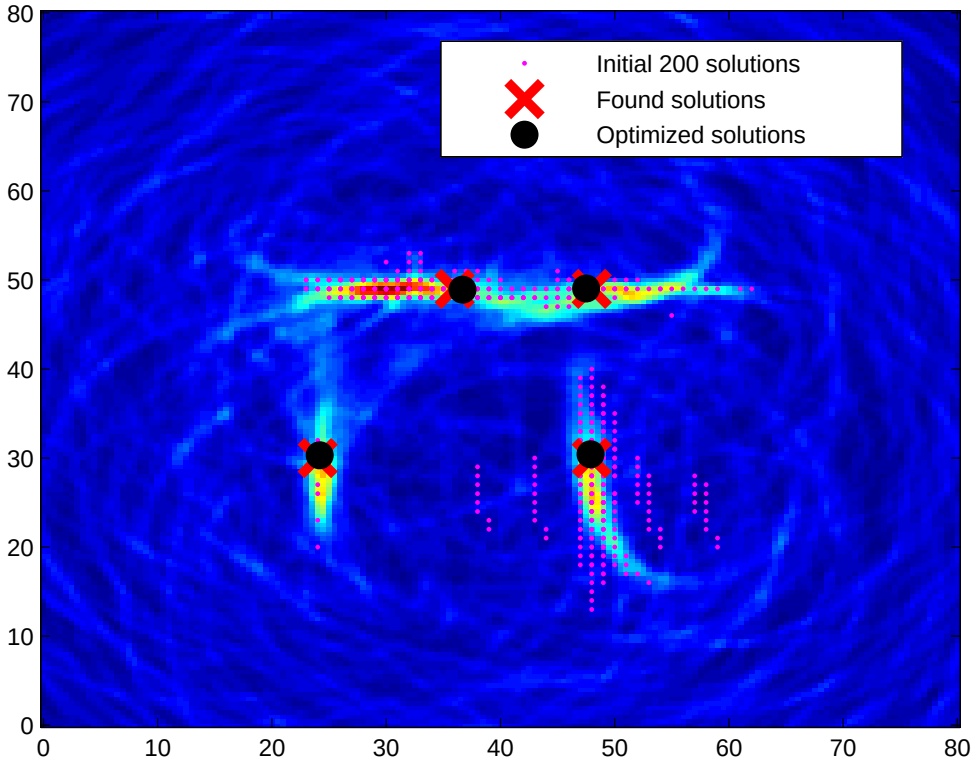

Fig. 2.6: Track initialization example. ML-PMHT LLR 2-D surface shown with top 200 initial solutions, final (four) found solutions, and optimized solutions.

initial target parameter set with the maximum LLR is selected — this vector is denoted as $\mathbf{x_{max\_1}}$, which is stored away as a potential track initialization point. Then, association probabilities corresponding to $\mathbf{x_{max\_1}}$ are computed for all the measurements. The association probability $w_j(i)$ for the single measurement $\mathbf{z}_j(i)$ is computed via (2.1.27); any measurements with an association probability of $w_j(i) > 0.1$ are deemed to come from the target at $\mathbf{x_{max\_1}}$, and these measurements are excised from the ML-PMHT batch. Now the LLRs are recomputed for the target parameter vectors that produced the LLRs in ς_0, but this time on the batch that had the measurements associated with $\mathbf{x_{max\_1}}$ excised. Denote this vector of (reduced) LLRs as ς_1. Now the ratio of the two is taken,

$$\Psi = \frac{\varsigma_1}{\varsigma_0} \tag{2.1.35}$$

Any target parameter vectors that correspond to a ratio $\Psi < 0.8$ are deemed to be from the same target as that described by $\mathbf{x_{max\_1}}$, and are removed. This is illustrated in Figure 2.4. The maximum LLR (the "Found solution") is caused by $\mathbf{x_{max\_1}}$ located at approximately $(x, y) = (50, 50)$. Now compare Figure 2.4 to Figure 2.3 — all the initial solutions (small dots) in the initial plot that are near the found solution ("x") have been excised.

Figure 2.5 shows the next step in the track initialization process. The solution corresponding to the next-highest LLR is found and saved (as $\mathbf{x_{max\_2}}$), and measurements associated with this solution are again excised from the batch. The

LLR ratio (2.1.35) is again computed, and any remaining solutions that have a value of $\Psi < 0.8$ are again discarded. In Figure 2.5, $\mathbf{x_{max\_2}}$ is located at approximately $(x, y) = (38, 50)$ — again, initial solutions in the vicinity of this have been discarded. This process continues until all unique solutions have been discovered. Next, each of these discovered solutions is optimized. Then, as a last check to ensure no duplicate tracks result, the optimized solutions (starting with the one producing the biggest LLR) are "reduced" in the same method as described above. Figure 2.6 presents the final result of the track initialization example — shown are the initial top 200 solutions, the reduced found solutions, and the final (reduced) optimized solutions. Lastly, the LLRs that correspond to the optimized solutions are compared to the tracking threshold. Tracks are initiated (and given an initial track health of 2) for any solutions with an LLR greater than the track threshold. For clarity, Figure 2.7 shows the overall track initialization logic.

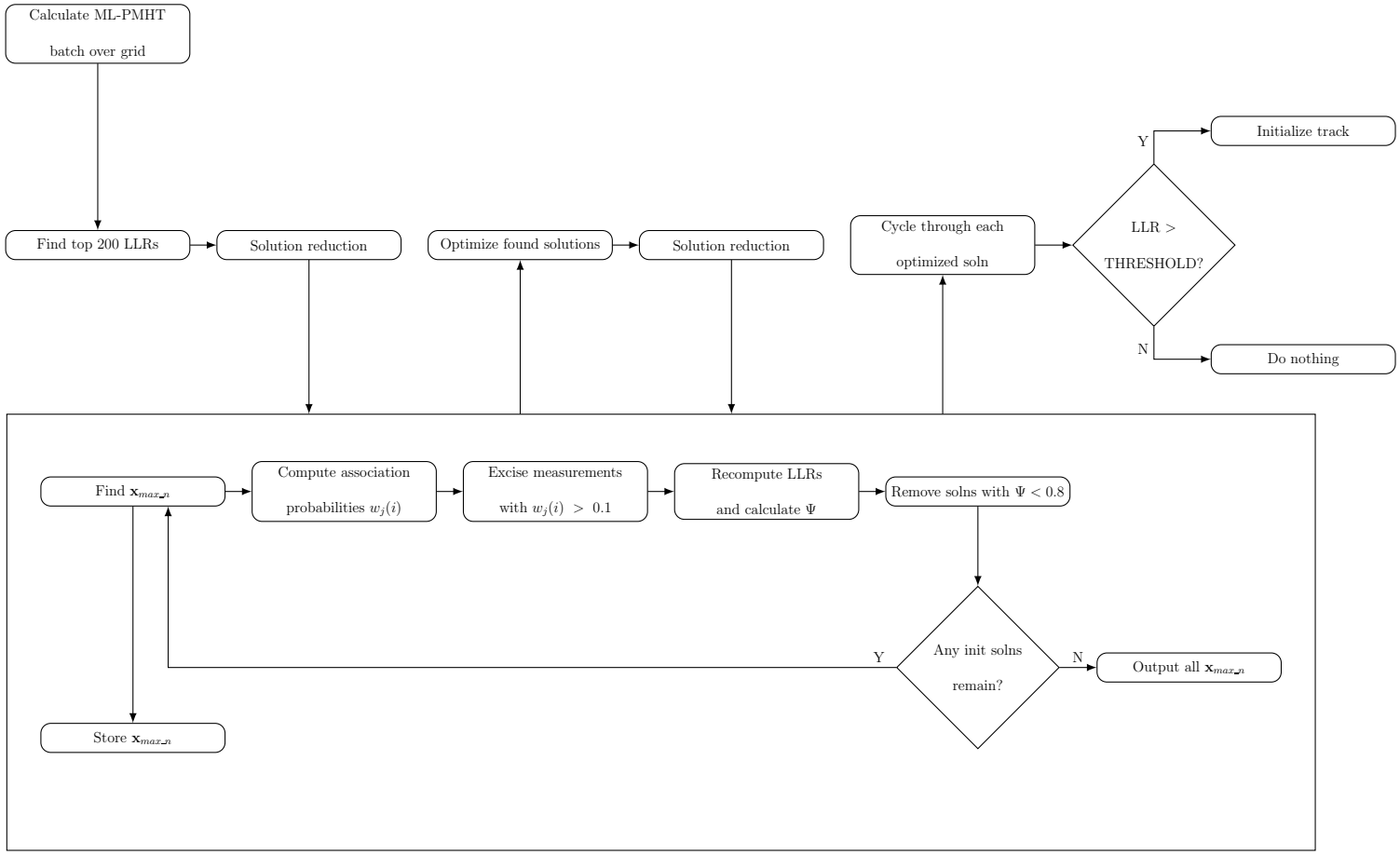

Fig. 2.7: Track initialization process

Chapter 3

Comparing ML-PDA and Multitarget ML-PMHT

3.1 Introduction

In Chapter 2, ML-PDA and ML-PMHT were presented as sequential single-target trackers, and in this form they had many similarities. Here, we start to explore the differences between the two algorithms.

This chapter first theoretically develops and implements ML-PMHT as a true multitarget tracker. As part of this, we find an expression for the Cramér Rao Lower Bound (CRLB) for ML-PMHT and show that ML-PMHT seems to be a *statistically efficient* estimator. As a result, this CRLB can be used as a covariance estimate for ML-PMHT. Moving on to Monte Carlo testing, we provide an example of the capability of the algorithms as very low observable trackers. (The term "low observable" is used to refer to targets with an SNR of 6-12 dB and "very low observable" to refer to targets with an SNR of less than 6 dB.) We then demonstrate that while ML-PMHT and ML-PDA have identical performance in single target cases, ML-PMHT is better than ML-PDA in multitarget cases or

when more than one measurement per scan is produced by the target.

This chapter will appear in the *Journal of Oceanic Engineering* as "ML-PDA and ML-PMHT: Comparing Multistatic Sonar Trackers for VLO Targets Using a New Multitarget Implementation."

3.2 Single target ML-PDA vs. ML-PMHT

The theory, assumptions and development of the single-target ML-PDA and ML-PMHT algorithms were presented in Chapter 2. In that chapter, the ML-PDA LLR for a batch of data was developed and (to review) is given by

$$\Lambda_0(\mathbf{x}, Z) = \sum_{i=1}^{N_w} \ln \left\{ 1 - P_d + \frac{P_d}{\lambda} \sum_{j=1}^{m_i} p[\mathbf{z}_j(i)|\mathbf{x}]\rho_j(i) \right\} \tag{3.2.1}$$

and the ML-PMHT LLR for a batch of data is given by

$$\Lambda_0^\dagger(\mathbf{x}, Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \left\{ \pi_0 + \pi_1 V p[\mathbf{z}_j(i)|\mathbf{x}]\rho_j(i) \right\} \tag{3.2.2}$$

These expressions will be used to compare the two algorithms and serve as a starting point for the development of the multitarget ML-PMHT LLR.

3.3 Relationship between ML-PDA and ML-PMHT

At first glance, the log-likelihood ratios for ML-PDA and ML-PMHT appear very dissimilar. However, the terms in the log-likelihood ratio for ML-PDA are actually just a subset of the terms in the log-likelihood ratio for ML-PMHT. Consider the

ML-PMHT likelihood (not log-likelihood) ratio for a single scan. This is written as

$$\frac{p_1^\dagger[Z(i)|\mathbf{x}]}{p_0^\dagger[Z(i)|\mathbf{x}]} = \prod_{j=1}^{m_i} \left\{ \pi_0 + \pi_1 V p[\mathbf{z}_j(i)|\mathbf{x}] \rho_j(i) \right\} \tag{3.3.1}$$

For any arbitrary number of measurements m_i, this equation multiplied out yields

$$\begin{aligned} \frac{p_1^\dagger[Z(i)|\mathbf{x}]}{p_0^\dagger[Z(i)|\mathbf{x}]} = \pi_0^{m_i} + \pi_0^{m_i-1}\pi_1 V \sum_{j=1}^{m_i} \mathcal{N}_j + \\ \pi_0^{m_i-2}\pi_1^2 V^2 \sum_{j_1=1}^{m_i-1} \sum_{j_2>j_1}^{m_i} \mathcal{N}_{j_1}\mathcal{N}_{j_2} + \cdots + \pi_1^{m_i} V^{m_i} \prod_{j=1}^{m_i} \mathcal{N}_j \end{aligned} \tag{3.3.2}$$

where the terms $\mathcal{N}_j$ represent $p[\mathbf{z}_j(i)|\mathbf{x}]$, the target-centered Gaussian terms. It should also be noted that the amplitude distribution ratios have been left out of the above equation — including them would not affect the following discussion. Equation (3.3.2) is the ML-PMHT likelihood ratio multiplied out for a single scan. It allows for more than one measurement to be assigned to a target — the terms with products of target-centered Gaussians account for this case. In the case of ML-PDA, the target assignment model allows only zero or one measurement to be assigned to the target. The first term in (3.3.2) accounts for the case of zero measurements being assigned to the target, and the second term in the equation accounts for the case of one measurement being assigned to the target. The other terms account for two or more measurements being assigned to the target. If the ML-PDA measurement model holds (or is at least very close to holding), then the product of the probabilities of two (or more) measurements being from a target is essentially zero and the terms with a product of two or more target-centered

Gaussians in the above expression go to zero. As a result, just the first and second terms are left. In this case, we can claim the following relationships. First

$$1 - P_d = \pi_0^{m_i} \tag{3.3.3}$$

Then,

$$P_d = m_i \pi_0^{m_i - 1} \pi_1 \tag{3.3.4}$$

and, since $E[m_i] = \lambda V$, one can take (approximately),

$$\pi_0^{m_i - 1} \pi_1 V = \frac{P_d}{\lambda} \tag{3.3.5}$$

If these relationships hold, then the ML-PDA likelihood ratio and the ML-PMHT likelihood ratio are equivalent.

3.4 Multitarget ML-PMHT

This section develops the multitarget formulation of ML-PMHT. It first discusses how it is necessary to implement ML-PDA for multiple targets in somewhat of an ad-hoc manner. Next, it presents the natural, multitarget version of the ML-PMHT LLR along with an elegant method of optimizing it. Then it develops a covariance estimate of the ML-PMHT state vector via the CRLB. Finally, it describes how to integrate these two concepts into a true multitarget tracking framework for ML-PMHT.

3.4.1 ML-PDA multitarget LLR

It is difficult to extend ML-PDA to multiple targets [18]. While it is technically possible to write the multitarget ML-PDA log-likelihood ratio, to take into account all the joint association events the number of terms increases rapidly with the number of targets, and the expression becomes practically intractable for any more than just a few targets. As a result, in order to handle multiple targets, the ML-PDA tracking framework treats scenarios with more than one target as a sequence of single-target problems (this process was described in Chapter 2). To briefly review, for a batch of data, it optimizes the single-target LLR (3.2.1), and if this value exceeds a certain threshold, a target is declared. (This threshold is determined by fitting the local maxima formed by the clutter to an extreme value distribution [17]. This topic will be explored in much more detail in Chapter 5.) Next, the measurement that has the highest association probability with that solution is excised from each scan, and the sequence is repeated for the next target. This method is not elegant, but it has been shown to work reasonably well for multitarget scenarios in [35] and [36].

3.4.2 ML-PMHT multitarget LLR

In contrast to ML-PDA, the ML-PMHT LLR is very easily extended to a multi-target framework. For K targets with state vectors $\mathbf{x}_1, \ldots, \mathbf{x}_K$, the multitarget

LLR is expressed as

$$\Lambda^{\dagger\dagger}(\mathbf{x}, Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \left\{ \pi_0 + V \sum_{k=1}^{K} \pi_k p[\mathbf{z}_j(i)|\mathbf{x}_k] \rho_{jk}(i) \right\} \tag{3.4.1}$$

where π_k is the probability that a given measurement is from the k^{th} target and $\sum_{k=0}^{K} \pi_k = 1$.

As discussed in Chapter 3, a benefit to using the ML-PMHT LLR formulation is that it is possible to optimize it with a closed-form expression using expectation-maximization (EM) [29]. The process of optimizing the ML-PMHT LLR for the single-target case was shown in Section 2.1.4 (as well as in [60]); here, we extend the optimization to the multitarget case. The multitarget version of the cost function is

$$J^{\dagger\dagger}(\mathbf{x}, Z) = \sum_{k=1}^{K} \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \left\{ [\mathbf{z}_j(i) - \mathbf{H}_k \mathbf{x}]^T \mathbf{R}_{ij}^{-1} [\mathbf{z}_j(i) - \mathbf{H}_k \mathbf{x}] + \ln(|2\pi \mathbf{R}_{ij}|) \right\} w_{jk}(i) \tag{3.4.2}$$

To briefly review, in the single target case, the ML-PMHT LLR can be optimized with EM as long as there exists a linear relationship between the predicted measurement $\hat{\mathbf{z}}$ and the state $\mathbf{x}$, where

$$\mathbf{x} = (x_0 \;\; \dot{x} \;\; y_0 \;\; \dot{y})^T \tag{3.4.3}$$

and the measurement matrix is

$$\mathbf{H} = \begin{bmatrix} 1 & t & 0 & 0 \\ 0 & 0 & 1 & t \end{bmatrix} \tag{3.4.4}$$

(For this to work, the measurements must obviously be in Cartesian space. While multistatic measurements and covariances are typically in bearing/time-delay space, they can be converted to Cartesian space following the work of [25] and [26].)

In the multitarget case for ML-PMHT, the state vector $\mathbf{x}$ becomes a $4K \times 1$ vector. (It is the vector in (3.4.3) stacked K times.) The association probability $w_{jk}(i)$ is

$$w_{jk}(i) = \frac{\pi_k p[\mathbf{z}_j(i)|\mathbf{x_k}]\rho_{jk}(i)}{\pi_0/V + \sum_{l=1}^{K} \pi_l p[\mathbf{z}_j(i)|\mathbf{x}_l]\rho_{jl}(i)} \tag{3.4.5}$$

The matrix $\mathbf{H}_k$ in equation (3.4.2) is

$$\mathbf{H}_k = \begin{bmatrix} \cdots & 0 & \cdots & & \cdots & 0 & \cdots \\ \cdots & 0 & \cdots & & \cdots & 0 & \cdots \\ & & & \mathbf{H} & & & \\ \cdots & 0 & \cdots & & \cdots & 0 & \cdots \\ \underbrace{\cdots \quad 0 \quad \cdots}_{4(k-1)\text{ columns}} & & & & \underbrace{\cdots \quad 0 \quad \cdots}_{4(K-k)\text{ columns}} & & \end{bmatrix} \tag{3.4.6}$$

This matrix is for the k^{th} target — the (inner) $\mathbf{H}$ is that given by (3.4.4) . There are $4(k-1)$ vectors of zeros to the left of $\mathbf{H}$ and $4(K-k)$ vectors of zeros to the right of $\mathbf{H}$. With these definitions, the solution to the problem is a simple vector quadratic minimization, and is expressed as

$$\mathbf{x} = \Big[\sum_{k=1}^{K} \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} w_{jk}(i) \mathbf{H}_k^T \mathbf{R}_{ij}^{-1} \mathbf{H}_k \Big]^{-1} \sum_{k=1}^{K} \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} w_{jk}(i) \mathbf{H}_k^T \mathbf{R}_{ij}^{-1} \mathbf{z}_j(i) \tag{3.4.7}$$

3.4.3 ML-PMHT Cramér-Rao lower bound

Here, the Cramér-Rao Lower Bound (CRLB) for ML-PMHT is developed, and it is shown that the ML-PMHT estimator is statistically efficient — i.e. the errors on the tracker state estimates meet the CRLB. Additionally, it turns out that the CRLB for ML-PMHT can be computed real-time; previous work on ML-PDA [41] showed that the CRLB for ML-PDA must be computed off-line with Monte-Carlo integration. Put together, these factors lead to a valuable result, namely, the CRLB calculation can be easily incorporated into the ML-PMHT tracking framework in order to represent the tracker error.

The full derivation for the ML-PMHT Fisher Information Matrix (FIM) and CRLB is presented in Appendix A; we summarize the key points in order to show why it is possible to compute the CRLB real-time for ML-PMHT (again, as opposed to ML-PDA). First, consider a window of ML-PMHT data with N_w scans. Since all measurements from scan to scan are assumed to be independent, the FIM $\mathbf{J}$ of the total window can just be written as the sum of the individual scans,

$$\mathbf{J} = \sum_{i=1}^{N_w} \mathbf{J}_i \tag{3.4.8}$$

After some, work (again, presented in Appendix A), a key point arises. The

expression for $\mathbf{J}_i$ is given as

$$\mathbf{J}_i = \sum_{j=1}^{m_i} \int_V \mathbf{D}_\phi^T \frac{\frac{[\pi_1 p_1^\tau(a_j)]^2}{|2\pi\mathbf{R}_j|} e^{-(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)} \mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}}{p(\mathbf{z}_j|\mathbf{x})^2} \times \mathbf{D}_\phi \prod_{l=1}^{m_i} p(\mathbf{z}_l|\mathbf{x}) d\mathbf{Z} d\mathbf{a} \quad (3.4.9)$$

Here, ϕ is the state-to-measurement conversion, and $\mathbf{D}_\phi$ is the Jacobian of ϕ. Next, $\mathbf{R}_j$ is the measurement covariance for the j^{th} measurement in the scan. (Since (3.4.9) only deals with a single scan, the i-subscripts are dropped.) Finally, $d\mathbf{Z} = \prod_{l=1}^{m_i} d\mathbf{z}_l$ and $d\mathbf{a} = \prod_{l=1}^{m_i} da_l$. All $p(\mathbf{z}_l|\mathbf{x})$ terms for $l \neq j$ are separable and integrate to one, and the term for $l = j$ is canceled by one of the terms in the denominator. All that is left within the integral is the single measurement $\mathbf{z}_j$. This reduces the calculation to a dimension equal to the number of dimensions in the measurement, plus one for the amplitude. This makes it possible for ML-PMHT to compute the FIM (and thus the CRLB) real-time as part of the tracking framework. In contrast, for the ML-PDA FIM calculation, a summation over all m_i measurements in the scan remains in the denominator [41]. Thus, the dimensionality of the calculation in this case is approximately m_i times the number of measurement dimensions. As a result, the ML-PDA FIM must be calculated off-line with Monte-Carlo integration.

After some work, the final expression for the ML-PMHT FIM is

$$\mathbf{J}_i = \mathbf{D}_\phi^T \sum_{j=1}^{m_i} \mathbf{G}_j^T \int_V \frac{\frac{[\pi_1 p_1^\tau(a_j)]^2}{|2\pi\mathbf{R}_j|} e^{-\xi_j^T\xi_j} \xi_j\xi_j^T}{\frac{\pi_0 p_0^\tau(a_j)}{V} + \frac{\pi_1 p_1^\tau(a_j)}{\sqrt{|2\pi\mathbf{R}_j|}} e^{-\frac{1}{2}\xi_j^T\xi_j}} d\xi_j da_j \frac{\mathbf{G}_j}{|\mathbf{G}_j|} \mathbf{D}_\phi \quad (3.4.10)$$

where ξ_j is a dummy variable of integration with dimensionality equal to the measurement dimensionality, and $\mathbf{G}_j$ is the Cholesky decomposition of $\mathbf{R}_j^{-1}$. This is inserted into (3.4.8), and then the CRLB is just taken as the inverse of the FIM. At this point, the efficiency of the ML-PMHT estimator was checked by using this FIM to calculate the normalized estimation error squared (NEES), which is given by

$$\text{NEES} = (\hat{\mathbf{x}} - \mathbf{x}_{truth})^T \mathbf{J} (\hat{\mathbf{x}} - \mathbf{x}_{truth}) \tag{3.4.11}$$

Scenario 1 (described in the sequel) was run 500 times, and for each run, the median NEES value for a valid track was saved (there were multiple NEES values for a run due to the sliding batch nature of the tracker implementation). The average of these NEES values came out to 4.15. The expected NEES for a 4-parameter state vector is 4, with a 95-percent confidence interval of [3.81, 4.19]. The result falls within this confidence interval, so the conclusion is that ML-PMHT is a statistically efficient estimator. A subset of 20 of these runs is shown in Figure 3.1 with the 95 percent position uncertainty ellipses at the beginning and end of the runs plotted. As expected, almost all of the solution points fall within their respective ellipses. The CRLB for ML-PMHT (as opposed to ML-PDA) can be computed real-time and then be used to accurately represent the error on the state estimate from the tracker.

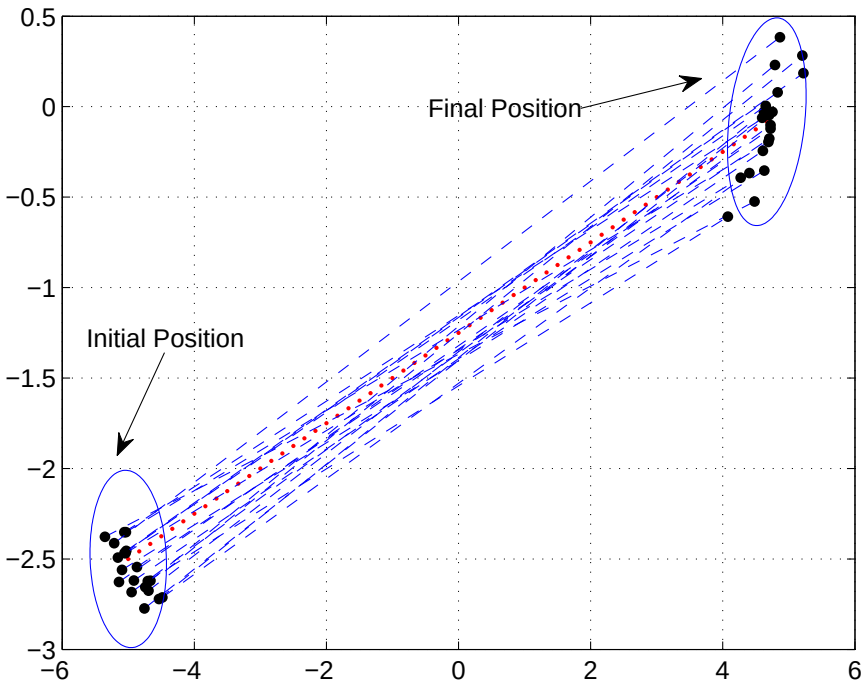

Fig. 3.1: CRLB 95 percent uncertainty regions for initial and final positions

3.4.4 ML-PMHT multitarget implementation

The ML-PMHT multitarget tracking framework was implemented in the track update sequence by testing all existing tracks for "closeness." Any tracks that were determined to be close were grouped together and optimized with the multitarget ML-PMHT log-likelihood ratio formulation in (3.4.1). Grouping tracks involved estimating the state covariance of all existing tracks. We let the CRLB developed above represent the state covariance $\mathbf{C}_n$ for the n^{th} target, and with this, a χ^2 test statistic [10] is evaluated for closeness between all possible pairs of active tracks in the form

$$S_{mn} = \Delta\hat{\mathbf{x}}_{mn}^T(\mathbf{C}_m + \mathbf{C}_n)^{-1}\Delta\hat{\mathbf{x}}_{mn} \tag{3.4.12}$$

where $\Delta\hat{\mathbf{x}}_{mn} = \hat{\mathbf{x}}_m - \hat{\mathbf{x}}_n$ (the difference of the state vector estimates between the m^{th} and n^{th} track). All track pairs with a statistic S_{mn} less than a given threshold are grouped together. More than two tracks could belong to a group by the use of this association — to join a group, an ungrouped track only had to meet the test described in (3.4.12) with one of the tracks already in the group.

3.5 Simulator and simulations

This section describes in detail the simulator that was used to create the data for the Monte Carlo runs and then presents the five scenarios used to compare ML-PDA and ML-PMHT.

3.5.1 Simulator description

The Monte Carlo runs were performed using a multistatic simulator developed at the University of Connecticut [75]. The simulator takes as input the source and receiver positions (all were stationary in this work), as well as the number of targets and target geometries — this information is all shown in Figures 3.5 through 3.14. Each transmitter generates a ping every 60 seconds, and for each source-receiver pair (i.e., a scan), the simulator calculates the number, position, amplitude, and Doppler of the clutter points, and the existence, position, amplitude, and Doppler of the target returns. (All pings in the simulation were treated as being Doppler-sensitive.)

Clutter measurement generation

First, to generate the clutter points, the simulator calculates the number of resolution cells that should be present. The number of resolution cells is treated as a function of σ_{az} and σ_t, the azimuthal and time-delay errors simulated for the system. The number of azimuthal resolution cells is approximated as

$$N_{az} = \frac{360}{BW} \tag{3.5.1}$$

where BW is the beamwidth of a receive beam in degrees. The simulator does not actually form any beams; to get a value of BW, it makes the assumption that the clutter is distributed uniformly in a beam, which gives $\sigma_{az} = BW/\sqrt{12}$. This

in turn leads to a final expression for the number of azimuthal resolution cells,

$$N_{az} = \left\lfloor \frac{360}{\sqrt{12}\sigma_{az}} \right\rfloor \tag{3.5.2}$$

(Here, "$\lfloor\ \rfloor$" signifies round towards the next lower integer.) For all of the scenarios in this work, the azimuthal uncertainty was $\sigma_{az} = 5^\circ$ (this and all other simulated error values are provided together in Table 3.2). Using similar logic, the number of delay (range) resolution cells is calculated with the expression

$$N_{delay} = \left\lfloor \frac{t_{ipt}/2 - t_{bt}}{\sqrt{12}\sigma_t} \right\rfloor \tag{3.5.3}$$

Here, t_{ipt} is inter-ping time — 60 seconds throughout this work, and t_{bt} is blanking time. This value is determined by the amount of time sound takes to travel from source to receiver. Finally, the number of resolution cells is simply given by the product of N_{az} and N_{delay}.

With the number of resolution cells determined, the clutter is generated. For each scan, to simulate the amplitudes of the clutter distribution, a number of random variables equal to the number of resolution cells is generated. For the first five scenarios, the amplitude of this clutter distribution was given a K-distribution. The probability distribution for the intensity of the clutter (i.e. the amplitude squared) is given by [1], [3]

$$f(p) = \frac{2}{\lambda\Gamma(\alpha_k)} \left(\frac{p}{\lambda_k}\right)^{(\alpha_k-1)/2} K_{\alpha_k-1}\left(2\sqrt{\frac{p}{\lambda_k}}\right) \tag{3.5.4}$$

where $p > 0$. Here, α_k is a K-distribution parameter ($\alpha_k = 0.5$ throughout this

work), $\lambda_k = 1/\alpha_k$, and K_ν is the Basset function (a modified Bessel function of the second kind) [4]. Random variables from the distribution shown in (3.5.4) are generated via [3]

$$K = -\Gamma\lambda_k \ln U \tag{3.5.5}$$

where U is a realization of a uniform random variable over the interval [0,1], and Γ is a realization of a Gamma random variable with shape parameter $\alpha = \alpha_k$ and a scale parameter $\beta = 1$.

For clarity, we note that the simulator generates "amplitude" log-likelihood ratios by operating in intensity (power) space — i.e. amplitude squared. Such log-likelihood ratios can be computed in either amplitude or intensity space; as long as the clutter and target distributions are consistent in terms of the units used, the LLR values will come out the same.

For the final scenario, the clutter amplitude was given a Rayleigh distribution following the work in [41], so the intensity of the clutter followed an exponential distribution. The clutter intensity in this case was simulated by generating exponential random variables with a mean of one.

Once the clutter realizations were created, they were thresholded. Amplitudes above the threshold were kept; amplitudes below the threshold were discarded. This threshold is what determined the clutter density; it was set so that the clutter density for the average source-receiver pair in the first five scenarios was

approximately 1×10^{-9} clutter points per unit volume; for the final scenario with Rayleigh clutter, the threshold was set so that the clutter density was approximately 1×10^{-7} clutter points per unit volume. The amplitude detector threshold values (in dB) are given in Table 3.1. After this, all remaining clutter points were

Table 3.1: Detector thresholds

Scenario	Threshold (dB)
1-4	12
5	14
1 (Rayleigh clutter)	2

given delay time and azimuth measurements by sampling from a uniform distribution over the proper time and azimuthal interval. Finally, all generated clutter measurements were given a Doppler component. The assumption was made that the clutter would be stationary; thus the Doppler measurements (simulated in the form of bistatic range rate), were generated from a zero-mean Gaussian with $\sigma_{\text{Doppler clutter}} = 2$ units/sec.

Target measurement generation

For every update, the target and source positions were updated according to user-defined course and speed segments. The expected target-reflected power at the

Table 3.2: Simulation errors

σ_{az} (deg)	σ_t (sec)	$\sigma_{\text{Doppler target}}$ (units/sec)	$\sigma_{\text{Doppler clutter}}$ (units/sec)
5	0.1	0.5	2

receiver was then calculated (in dB) with

$$P_{rec} = SL_{1000} - 10\log_{10}(R_{ST}/1000) - 10\log_{10}(R_{TR}/1000) + 10\log_{10}(TS) \quad (3.5.6)$$

Here, SL_{1000} is the signal power 1000 distance units from the target (all distance units in this simulation are arbitrary), R_{TS} is the distance from the target to source, R_{TR} is the distance from the target to the receiver, and TS is the bistatic target strength. This bistatic target strength is a function of the bistatic angle, defined in [26]. For this simulation, the value of TS as a function of bistatic angle is shown in Figure 3.2. (Note that with (3.5.6) we are not trying to implement directly the physical effects described by the active sonar equation. Instead, we are trying to implement a target amplitude model that has a known dependency on range and bistatic aspect and is given a reference level at a convenient, fixed distance from the target.) Finally, the "measured" target amplitude was simulated having a Rayleigh distribution, with expected amplitude given by (3.5.6). The

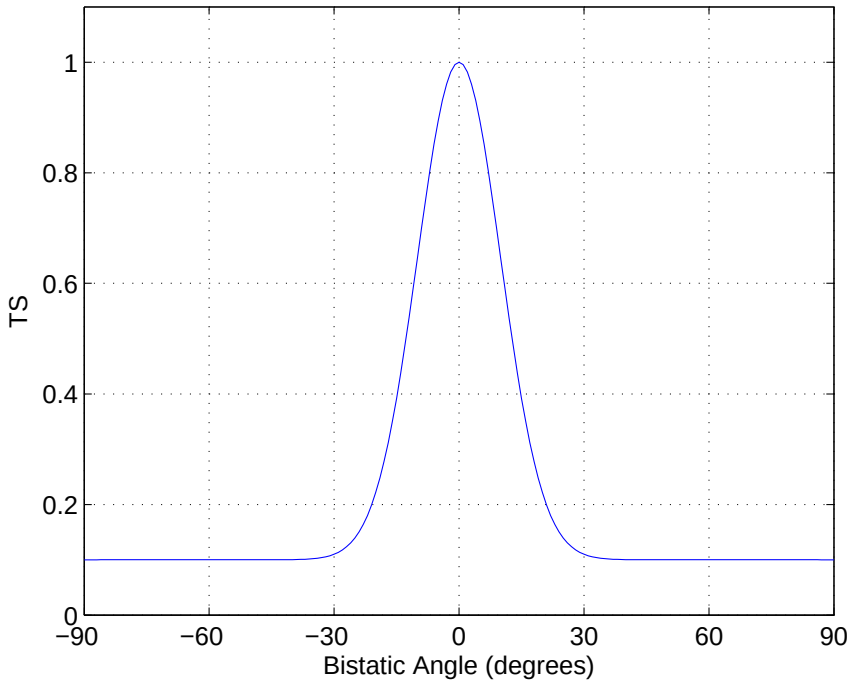

Fig. 3.2: TS as a function of bistatic angle

measured target amplitude random variable is simulated (in dB space) with

$$P_{tgt\ meas} = P_{rec} + 10\log_{10}(-\ln(U)) \tag{3.5.7}$$

Here, U is a uniform random variable in the interval [0,1]. (It is easy to show that if $P_{tgt\ meas}$ is converted to intensity space, (3.5.7) produces a realization of an exponential random variable with mean $10^{P_{rec}/10}$.) If target return power $P_{tgt\ meas}$ is greater than the respective threshold for its scenario shown in Table 3.1, then the (target-originated) detect is kept in the data for processing. The values of SL_{1000} are shown in Table 3.3.

Finally, the time, azimuth, and bistatic range-rate values for all target measurements are calculated and then corrupted with noise. Based on the (deterministic) positions of the source, receiver, and target, the delay time t and the azimuth θ (from the receiver to the target) are determined via simple geometry. These values of t and θ are corrupted by adding realizations of zero-mean Gaussian random variables with variances σ_t^2 and σ_{az}^2 to them, respectively. The bistatic range-rate is calculated following the work of [26], and then this is corrupted with a zero-mean Gaussian random variable with variance $\sigma^2_{\text{Doppler target}}$.

ML-PMHT target measurement generation model

The majority of simulations in this work followed the ML-PDA target measurement generation model — that is, at most one measurement originated from the

Table 3.3: SL_{1000} values as a function of scenario

Scenario	1	2	3	4	5	1 (Rayleigh clut)
SL_{1000} (dB)	35	40	45	40	45	27

target in a given scan (this is implicit in the simulator discussion above). However, the simulator was also set up with the ability to generate multiple target measurements per scan — i.e. the ML-PMHT target measurement generation model. This was done by assuming P_d is known, and then writing the probability mass function (PMF) describing the number of target measurements $f(x, \lambda_t)$ as

$$f(x, \lambda_t) = \begin{cases} 1 - P_d & x = 0 \\ \frac{P_d}{1-e^{-\lambda_t}} \frac{e^{-\lambda_t} \lambda_t^x}{x!} & x > 0 \end{cases} \tag{3.5.8}$$

Let $N = E[x]$, the expected number of measurements from the PMF. Calculating the expected value of (3.5.8) produces the following relationship

$$N(1 - e^{-\lambda_t}) = P_d \lambda_t \tag{3.5.9}$$

Again, the value of P_d is assumed known (in this work, it was $P_d \approx 0.8$). Then, (3.5.9) can be numerically solved for λ_t as a function of N. Results of this are presented in Table 3.4. To produce, on average, N target-generated measurements per scan, the simulator first determines if the contact power is greater than the threshold, as described above. If the target power does exceed the threshold,

Table 3.4: Values for λ_t as function of expected number of target measurements

N	λ_t
1	0.4642
2	2.2316
3	3.5628

the simulator draws a realization from a truncated Poisson PMF with parameter λ_t, and creates this many (target) measurements, again, in the same manner as described above.

3.5.2 Scenarios

Five benchmark multistatic sonar scenarios, initially developed in [60], were used to measure performance differences between ML-PDA and ML-PMHT with Monte Carlo testing. Each scenario was designed so target detections from a single source-receiver pair would be present approximately 80 percent of the time in a given scan, and clutter was set at a level that made the problem as difficult as possible while not slowing down run times to the point of precluding Monte Carlo testing. The various scenario parameters are listed in Table 3.5 (these parameters, used in the simulation, were also matched in the actual ML-PDA and ML-PMHT tracking code). All scenarios are shown (with example results overlaid) in Figures

3.5 – 3.16. Each of the scenario target geometries was designed for a specific test purpose. Then, given these target geometries, the sensors were laid out in an effort to obtain an average target P_d of 0.8 in any given scan. (The one exception to this occurs in Scenario 5 and is described below.) The pinging strategy was purposely kept very simple – every transmitter had a (simulated) ping every 60 seconds. The scenarios are described as follows:

Baseline scenario

Scenario 1 featured a single target moving in a straight line past a source and a receiver (the source was a receiver as well). This was intentionally created as one of the simplest possible multistatic scenarios. As is shown in Section 3.2, ML-PDA and ML-PMHT should converge to the same value under benign conditions, and this scenario is a case where this convergence should hold true.

Close targets with similar dynamics scenario

Scenario 2, shown in Figure 3.7, features three targets very close together (they have a separation of only 500 distance units) with similar motion dynamics – i.e. they are maneuvering in a coordinated fashion. This makes it very difficult for any algorithm to distinguish and separate targets from each other. This will test if the multitarget implementation for ML-PMHT is an improvement over the sequential single-target implementation of ML-PDA.

Close targets with different dynamics scenario

Scenario 3, shown in Figure 3.9, has two targets closing each other from the east and west. During the middle of the scenario, the targets will be in close proximity to each other, but they will have different motion dynamics — at the time they are close, they will be moving in opposite directions. Again, this is designed to test each tracker's ability to follow close targets, but under slightly easier circumstances than in scenario 2.

Large number of targets scenario

Scenario 4, shown in Figure 3.11, features ten very low-speed targets and three relatively high-speed targets. This scenario was designed simply to measure each algorithm's ability to track a (relatively) large number of targets, especially when some of those targets are low-speed and exhibit very low Doppler.

Switching targets scenario

Scenario 5, shown in Figure 3.13, is similar to scenario 2 in that it has two targets in close proximity to each other with the same motion dynamics. However, the targets start out with a distinct separation and then close on each other. At the point where the targets are about to intersect, they turn and parallel each other. Such motion, with the targets approaching each other (where their projected motion has them crossing), will make it very difficult for any trackers to continually

associate measurements with the correct targets and not to switch targets. Additionally, this scenario was given a relatively high number of receivers (16), and due to the geometry, most source receiver pairs will have a P_d of much less than 0.8 — many scans will just have clutter measurements. This is effectively raising the clutter levels that the tracker will see.

These scenarios were all run using the ML-PDA target measurement generation model, where the target produced at most one measurement per scan. Scenario 1 was then re-run with the ML-PMHT target measurement generation model, where more than one measurement was allowed to be generated by the target in a scan. This scenario was run three times following the target measurement generation method described in Section 3.5.1; the expected number of target measurements per scan was one, two and three, respectively. (Note the case of one target measurement per scan is not the ML-PDA target measurement generation model. In this case, on average the target generates one measurement per scan, but there could be more. In the ML-PDA case, the number of target measurements per scan is strictly limited to zero or one.)

At the conclusion of each scenario, the following metrics were evaluated: target in-track percentage, target position root-mean square error (RMSE), track fragmentation, target duplicate tracks, overall number of false tracks, and average false track length. These metrics were calculated in the following manner. First,

Table 3.5: ML-PMHT and ML-PDA parameters

	ML-PMHT			ML-PDA	
	π_0	π_1	V	λ	P_d
Scen. 1-4	0.95	0.05	1.26×10^9	3.7×10^{-9}	0.8
Scen. 5	0.95	0.05	1.26×10^9	6.7×10^{-10}	0.8
Scen. 1 Rayleigh clut.	0.95	0.05	1.26×10^9	2.0×10^{-7}	0.8

tracks were associated with either a target or clutter. A track was associated with a target if its normalized estimation error squared (NEES) value was less than 10 for scenarios with a single target or widely-spaced targets. (In most situations, as is discussed above, the estimators were statistically efficient.) For scenarios with closely-spaced maneuvering targets, the CRLB tended to underestimate the covariance (the estimators were not statistically efficient here), so the NEES was not as reliable in associating tracks to targets. Instead, in these cases, a track was associated with a target if the average distance between track points and their associated truth points was less than 2000 distance units. If a track could be associated with more than one target, then it was assigned to whichever one was closest. Target in-track percentage was then calculated by taking the ratio of

target truth points that had a track associated with them to the total number of target truth points. Target duplicate tracks were calculated by simply counting the number of targets that had more than one track associated with them at a given time. The root mean square error was then calculated for all tracks that were associated with targets. This RMSE value was only calculated for a target on points where there was both an ML-PDA track and an ML-PMHT track associated with it. This was done so as not to penalize one algorithm that was barely holding track (with a resultant poor RMSE) while the other algorithm was not tracking at all. Track fragmentation was determined by counting the number of breaks in track for a target. Finally, the overall number of false tracks was simply the number of tracks that were not associated with a target, and the mean false track length was the average number of updates for which a false track was active. For each of the Monte Carlo sets, 200 runs were performed. Additionally, the confidence intervals shown in Tables 3.6 through 3.12 are at the 95 percent level.

3.6 ML-PDA vs. ML-PMHT performance comparisons

Both ML-PDA and ML-PMHT showed themselves to be excellent trackers for targets with low amplitude returns (detailed below). First, to get an idea of the level of difficulty of the problem, all measurements from a batch are plotted in Figure 3.3 for scenario 1 with the Rayleigh clutter distribution (the scenario with

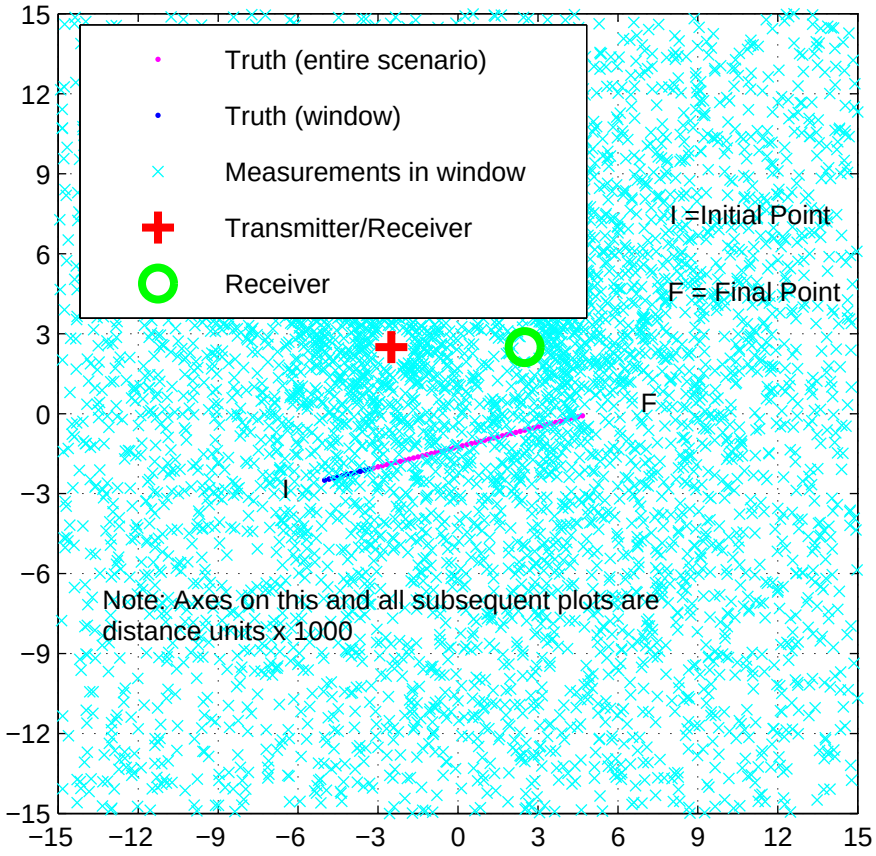

Fig. 3.3: All measurements from a window of data (11 update periods) for scenario 1 with Rayleigh clutter

the highest level of clutter).

Using this same scenario, in order to provide a comparison to other algorithms with known performance, in-track percentage as function of SNR was calculated. This is shown in Figure 3.4 for ML-PDA (results were practically identical for ML-PMHT). For this example (in contrast to the rest of the runs), expected target SNR was fixed, starting at 4 dB with a 2 dB measurement threshold. The

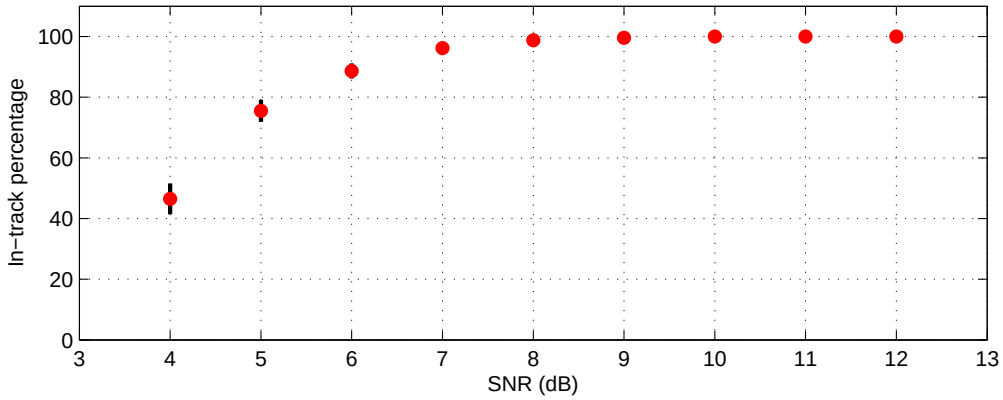

Fig. 3.4: ML-PDA in-track percentage vs. expected target SNR for scenario 1 with Rayleigh clutter

expected target SNR was then moved up in 1 dB increments, with Monte Carlo simulations being run at each SNR level. Note that the tracker achieves a P_{DT} (probability of track detection/in-track percentage) of approximately 50 percent at a target SNR of only 4 dB, and a P_{DT} of about 90 percent at 6 dB. By a target SNR of 8 dB, the tracker essentially has a P_{DT} of 100 percent. This shows that ML-PDA and ML-PMHT are effective very low observable trackers.

After this, ML-PDA and ML-PMHT were run on all five scenarios in their respective multitarget tracking modes. ML-PDA operated in the sequential single-target mode, while ML-PMHT was run in its true multitarget mode for tracks that were close together.

All the scenarios except one featured clutter that had K-distributed amplitudes.

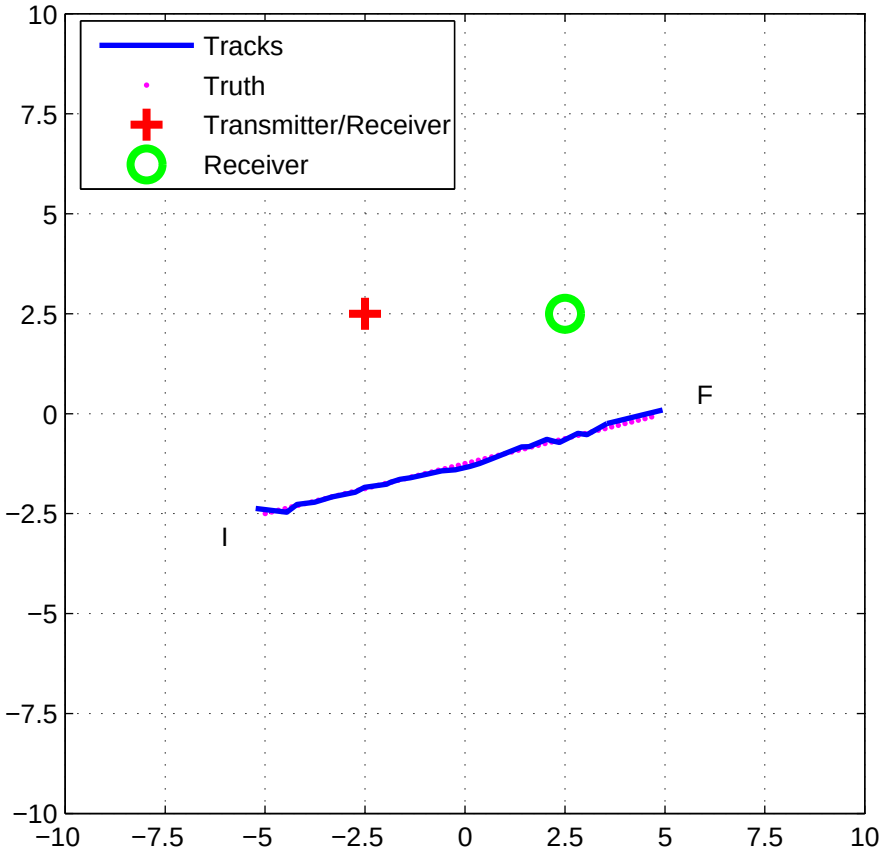

Fig. 3.5: Scenario 1 (baseline) and ML-PDA sample estimated track in one run

The high-clutter version of scenario 1 featured Rayleigh-distributed clutter amplitudes. In all cases, the feature LLRs calculated by ML-PDA and ML-PMHT matched the correct clutter distributions.

For scenarios with a single target or easily distinguished targets, ML-PDA and ML-PMHT had virtually identical performance over the Monte Carlo runs. As is shown in Section 3.3, the ML-PMHT log-likelihood ratio should be very close to the ML-PDA log-likelihood ratio, so the tracking results of the two should be

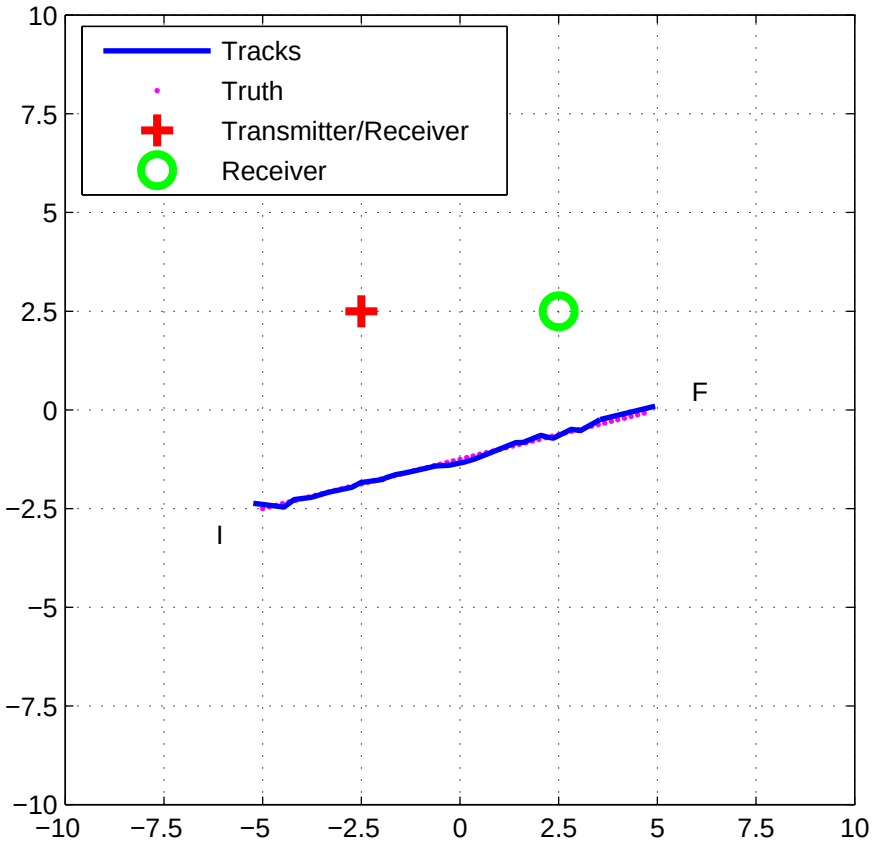

Fig. 3.6: Scenario 1 (baseline) and ML-PMHT sample estimated track in one run

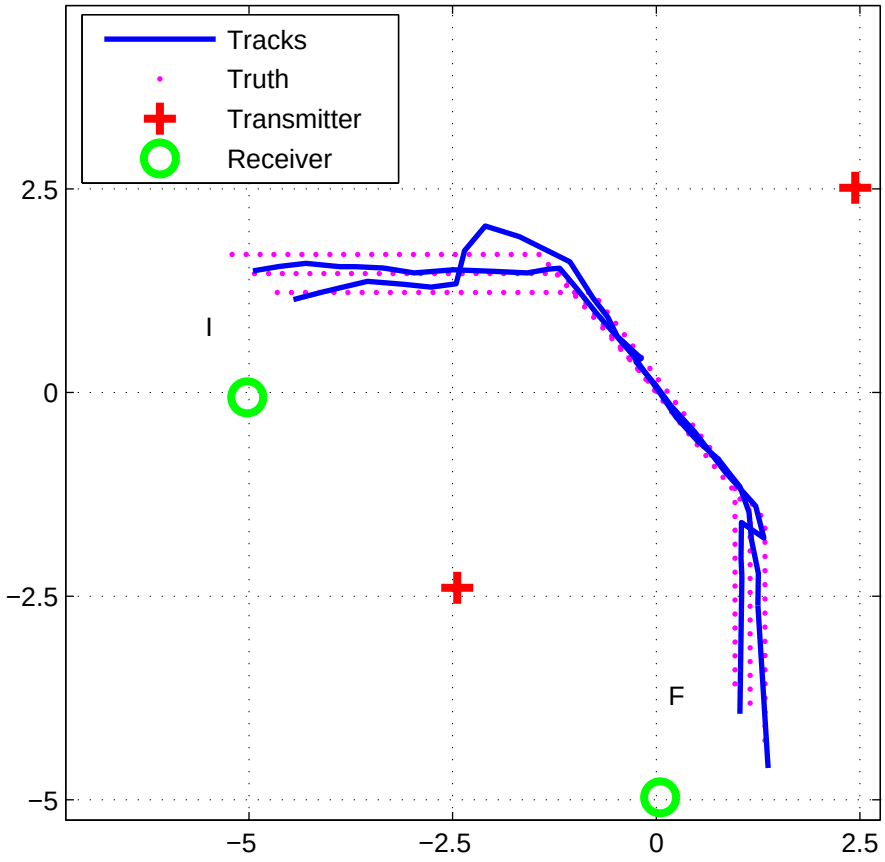

Fig. 3.7: Scenario 2 (three close targets with similar dynamics) and ML-PDA sample estimated tracks in one run

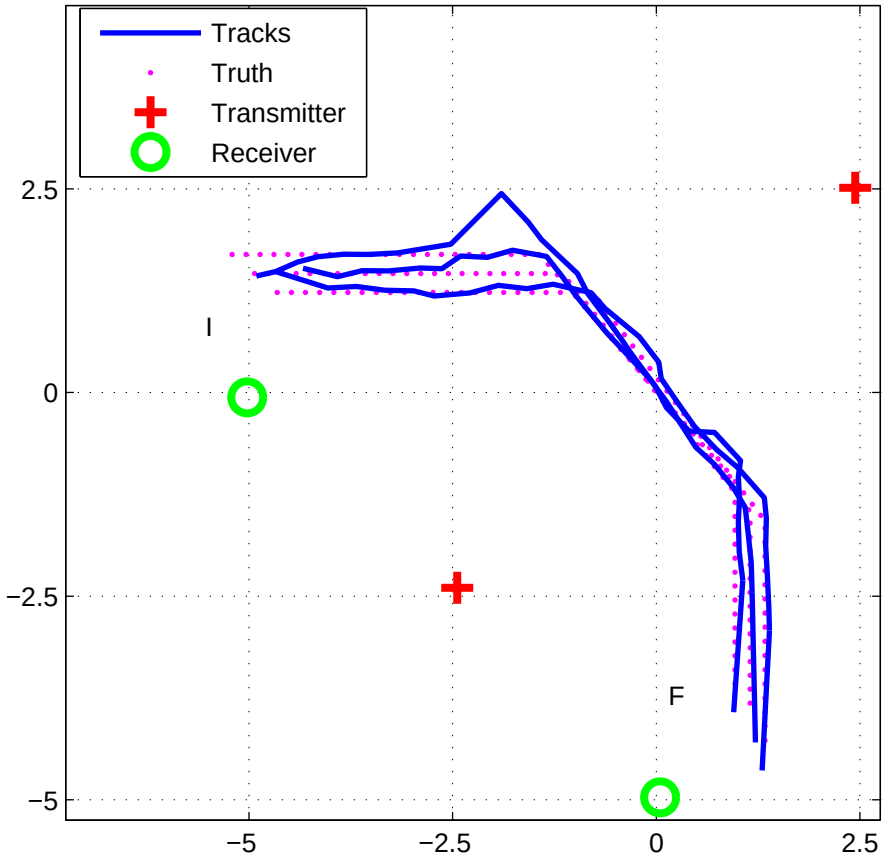

Fig. 3.8: Scenario 2 (three close targets with similar dynamics) and ML-PMHT sample estimated tracks in one run

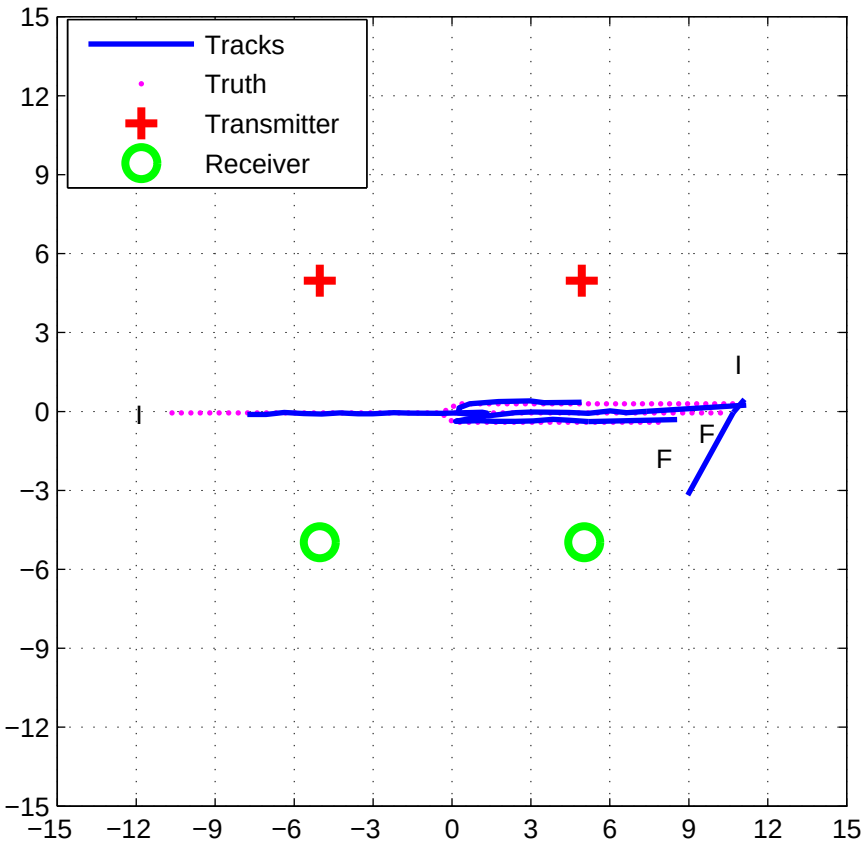

Fig. 3.9: Scenario 3 (two close targets with different dynamics) and ML-PDA sample estimated tracks in one run

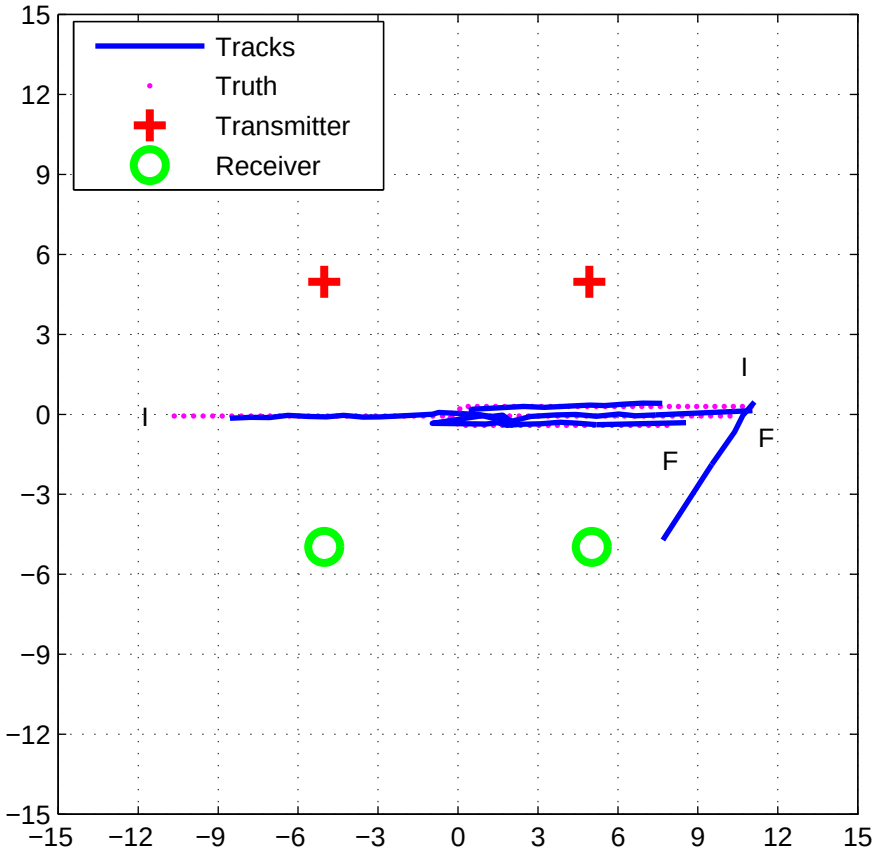

Fig. 3.10: Scenario 3 (two close targets with different dynamics) and ML-PMHT sample estimated tracks in one run

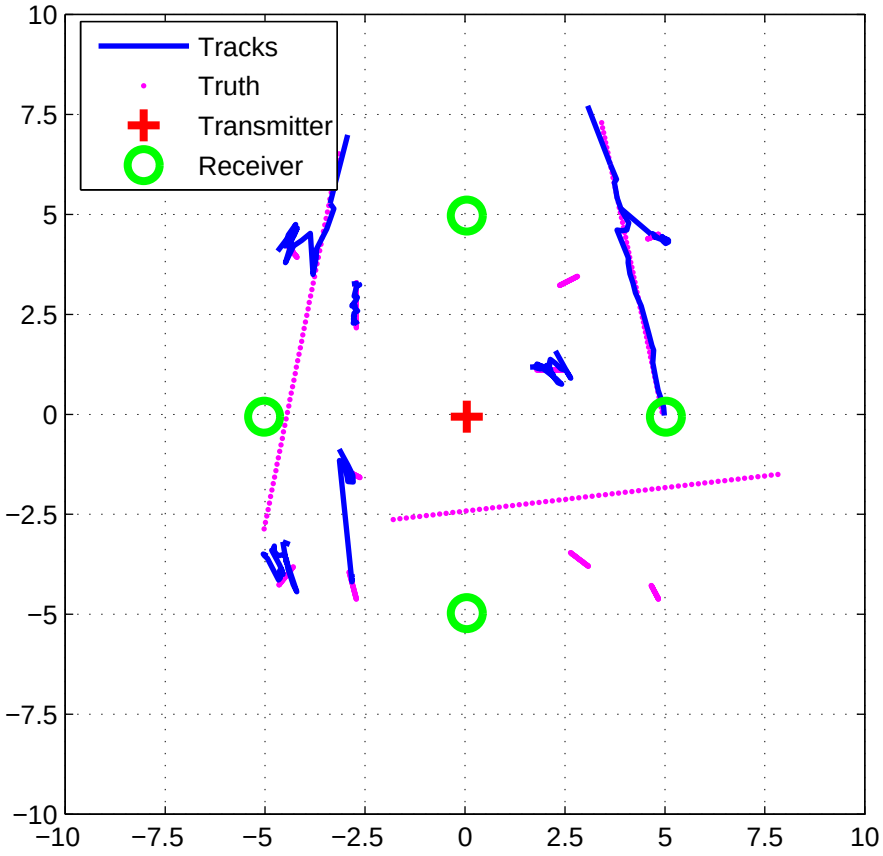

Fig. 3.11: Scenario 4 (large number of targets) and ML-PDA sample estimated tracks in one run

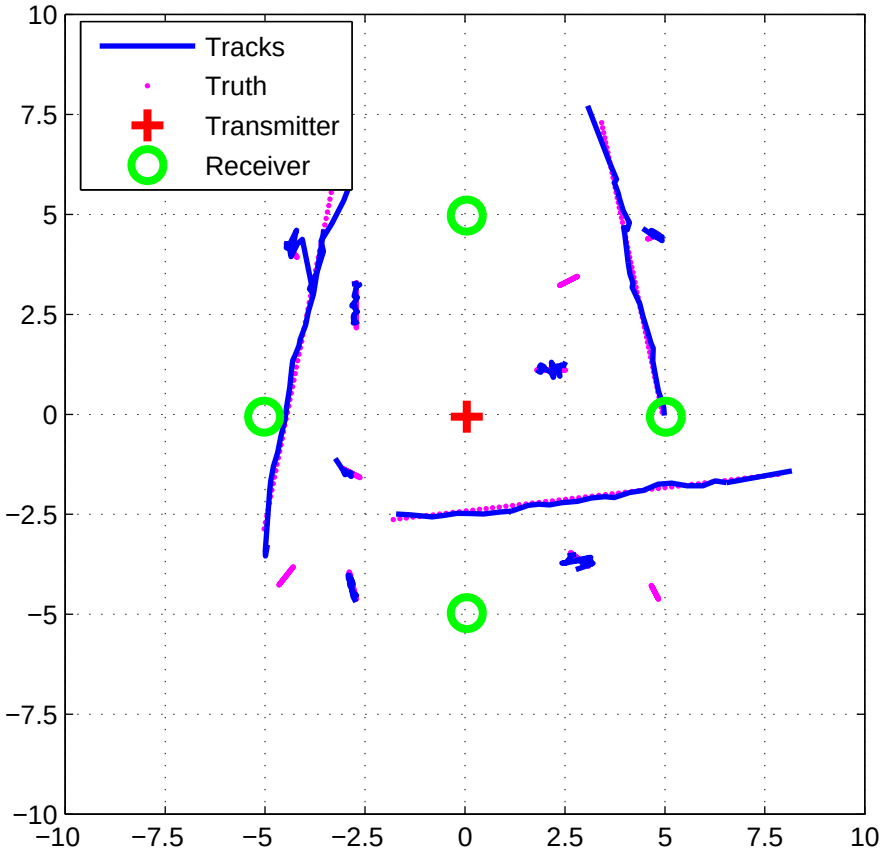

Fig. 3.12: Scenario 4 (large number of targets) and ML-PMHT sample estimated tracks in one run

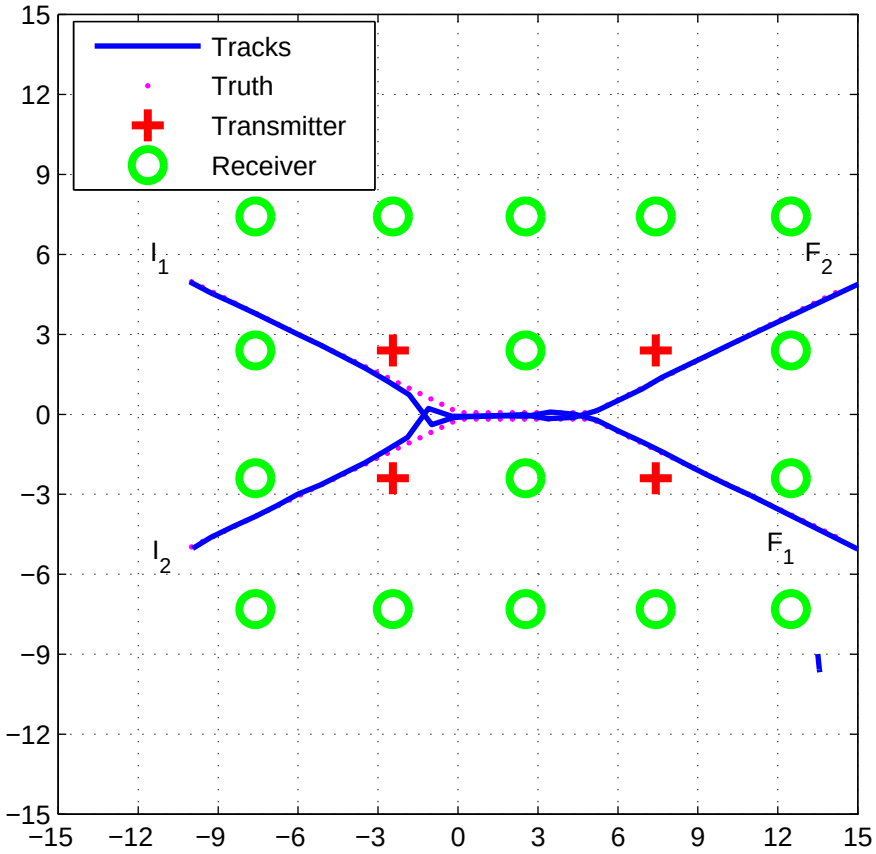

Fig. 3.13: Scenario 5 (switching targets) and ML-PDA sample estimated tracks in one run (switch occurred)

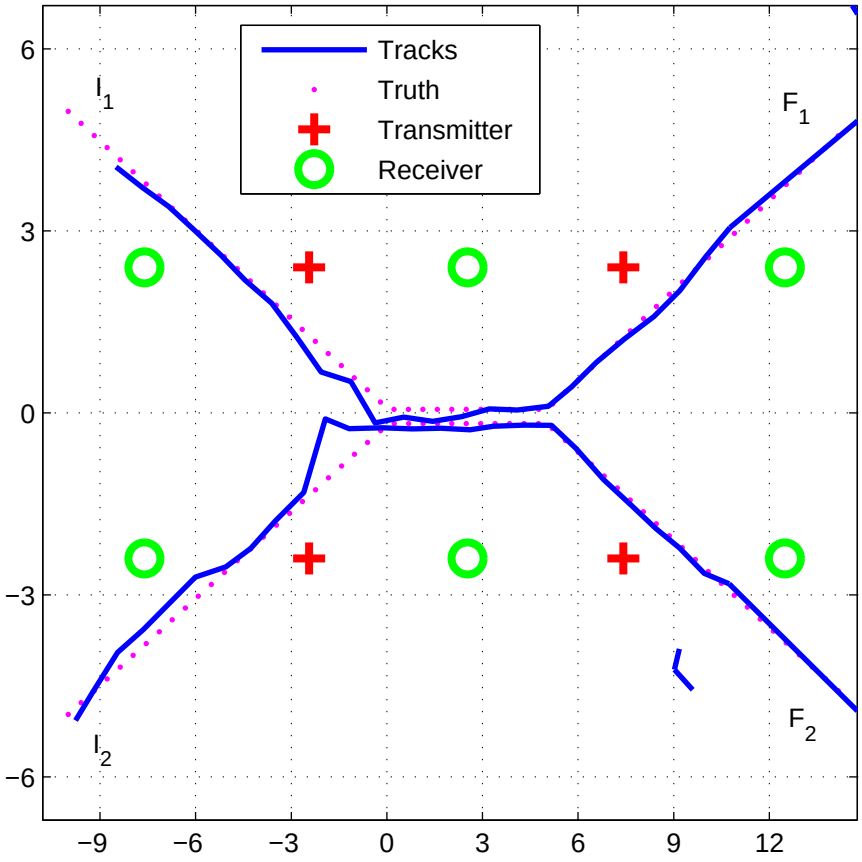

Fig. 3.14: Scenario 5 (switching targets) and ML-PMHT sample estimated tracks in one run, zoomed-in view (no switching occurred)

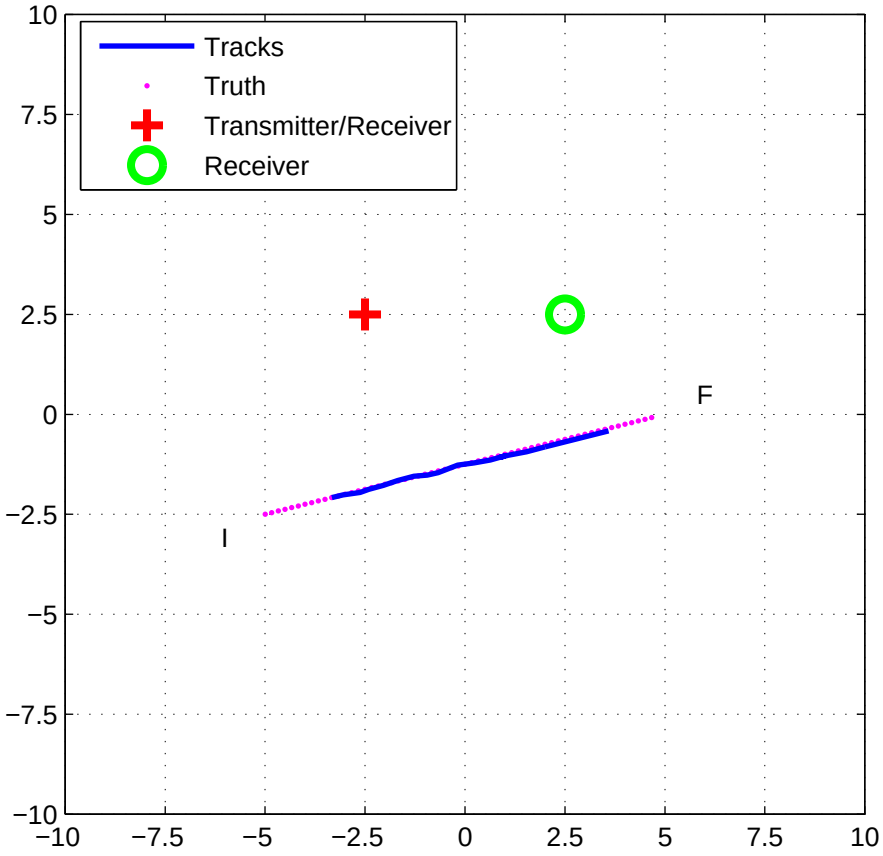

Fig. 3.15: Scenario 1 (baseline) and ML-PDA sample estimated tracks with high density Rayleigh clutter in one run

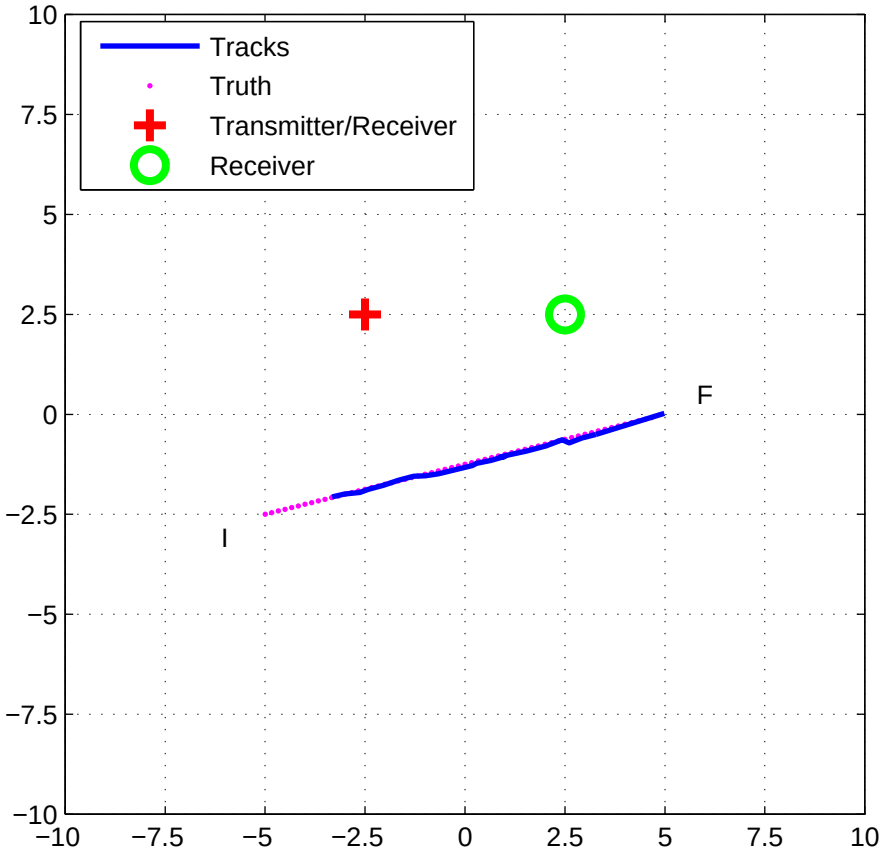

Fig. 3.16: Scenario 1 (baseline) and ML-PMHT sample estimated tracks with high density Rayleigh clutter in one run

similar. The terms in (3.3.2) that represent more than one measurement in a scan originating from the target are clearly very close to zero, resulting in the ML-PDA and the ML-PMHT LLRs being nearly identical and thus producing nearly identical tracking results.

Individual examples taken from the Monte Carlo runs help to further illustrate this. For scenario 1, an ML-PDA result shown in Figure 3.5 and an ML-PMHT result shown in Figure 3.6 are virtually indistinguishable from each other. For the same scenario with the Rayleigh clutter, results for the two algorithms (Figure 3.15 for ML-PDA and Figure 3.16 for ML-PMHT) are also very similar. Finally, for scenario 3, the ML-PDA example in Figure 3.9 and the ML-PMHT example in Figure 3.10 appear very much alike.

For the scenarios with targets in close proximity to each other with similar motion dynamics (scenarios 2 and 5), or the scenario with multiple targets that were close enough to potentially cause a tracker to switch targets (scenario 4), the results in Tables 3.6 – 3.11 show that ML-PMHT clearly outperforms ML-PDA. This is due to the true multitarget implementation for ML-PMHT in contrast to the sequential single-target tracking logic that is necessary for ML-PDA. In scenario 2, ML-PMHT is able to track targets 1 and 2 in excess of 80 percent of the time and track target 3 nearly 50 percent of the time. In contrast, ML-PDA is tracking target 2 (the middle target) 100 percent of the time, and targets 1 and 3 (the

outer targets) less than 10 percent of the time. (Overall, ML-PMHT is tracking at least one target 100 percent of the time.)

The reason ML-PDA is not performing as well as ML-PMHT in this case is the sequential single-target implementation that is necessary for ML-PDA. Figure 3.17 illustrates what is happening in scenario 2. This figure shows measurements from three targets in close proximity to each other, all moving in the same direction. For clarity of illustration, the measurements have no noise, and there are no clutter measurements shown. ML-PDA is implemented in a sequential mode; the first track "finds" the high-SNR measurement in each scan, and then these measurements are excised from the data. The next track finds the remaining (relatively) high-SNR measurement, and again, these measurements are excised from the data. Since both found tracks are using data from all three true targets, the tracks tend to zigzag back and forth over the middle target (and each other). As a result, the track scoring will most likely result in the middle target being tracked by both tracks. Over many trials, this will produce a high P_{DT} value for the middle target, a duplicate track for the middle target, and low P_{DT} for the outer targets. This is exactly what is seen in the Monte Carlo results for scenario 2 — for ML-PDA the middle target had a P_{DT} of 100 percent, while the outer targets had a P_{DT} of less than 10 percent. The middle target also had, on average, 1.07 duplicate tracks associated with it.

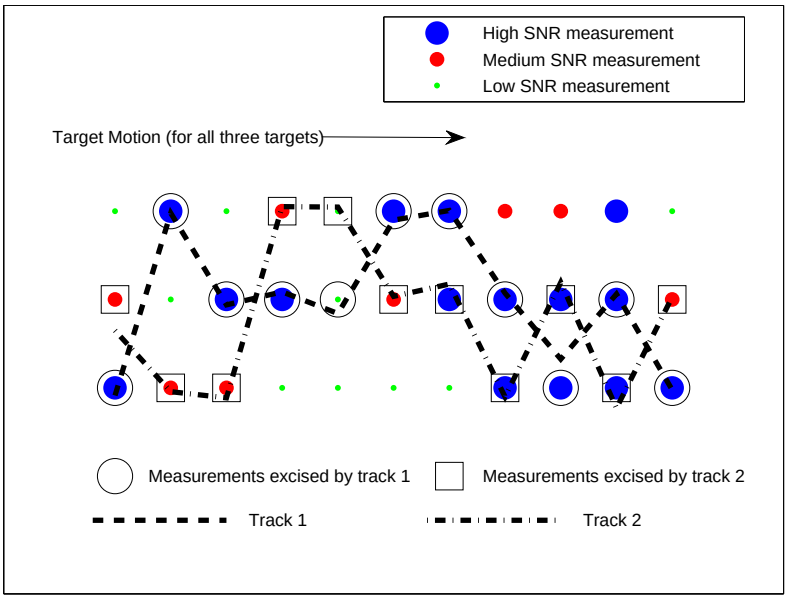

Fig. 3.17: Example of multitrack measurement assignment for ML-PDA

In contrast, ML-PMHT, with its more natural and appealing true multitarget formulation, can better find all three targets. Simultaneously optimizing for multiple targets at once prevents the "claiming" of the high-SNR measurement by the first track to run through the data; instead, the high-SNR measurements are more equally (and correctly) divided between the three tracks, as is shown in Figure 3.18. Scenario 5 showed similar results. ML-PDA was only able to track both targets approximately 45 percent of the time, while ML-PMHT was able to track both targets approximately 65 percent of the time. This number is actually slightly misleading; the relatively low numbers for ML-PDA are due to the fact that tracks using this algorithm switched targets on the portion of the trajectory

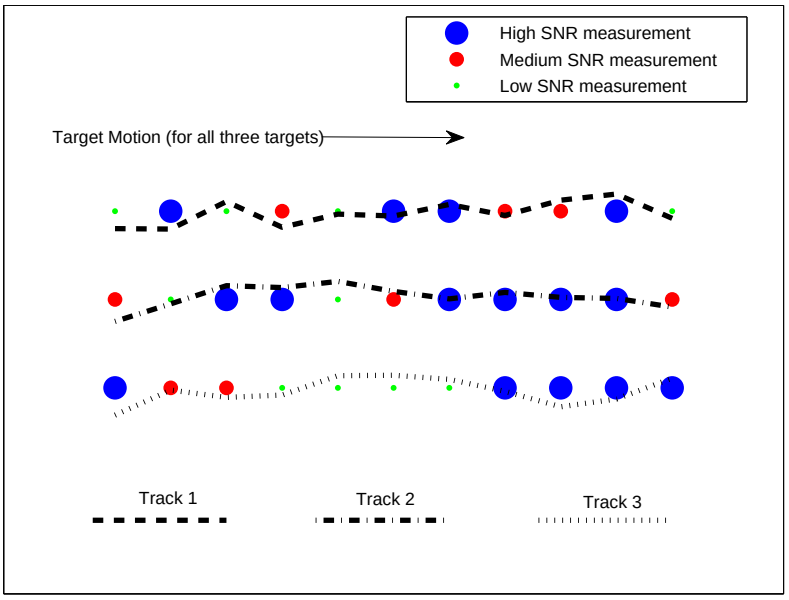

Fig. 3.18: Example of multitrack measurement assignment for ML-PMHT

where the targets were close, and the tracks ended up on the wrong targets, driving down P_{DT}. There was some switching for ML-PMHT as well, but as is seen by the results, ML-PDA was more susceptible to this problem. This is a result of the sequential single-target implementation used for ML-PDA that is described above.

Individual examples from the Monte Carlo runs again reinforce this conclusion. A scenario 2 result for ML-PDA is shown in Figure 3.7. Here, the first track (on the middle target) "claims" and then excises all the high-SNR measurements. A second track is initiated, and it claims the remaining (relatively) high-SNR measurements, in the process crossing over the first track several times. As a

result, it appears that this track would be associated with the middle target as well. There are not enough high-SNR measurements left to form a third track. In contrast, the multitarget ML-PMHT tracker, shown in Figure 3.8 is able to track all three targets with a minimum of switching. Examples from scenario 5 show similar results. In Figure 3.13, ML-PDA is not able to separate the targets on the portion of the track where the two targets are paralleling each other at very close range; during this portion, the tracks actually switch between targets five times. In contrast, in Figure 3.14, the multitarget ML-PMHT tracker is able to track both targets without any switching.

Finally, ML-PMHT outperformed ML-PDA on scenario 4 for almost all of the targets for similar reasons. (Results from only targets 7 and 12 are shown below; they are fairly representative of all 13 targets in this scenario). Again, the sequential single target tracking framework for ML-PDA could not perform as well as the true multitarget implementation for ML-PMHT. Tracks using the ML-PDA implementation were far more susceptible to being drawn off from one target to another by high-SNR measurements (similar to the effect illustrated in Figure 3.17). Compare the performance of ML-PDA on this scenario in Figure 3.11 with the performance of ML-PMHT in Figure 3.12. Several instances of track switching are visible in the ML-PDA plot that are not seen in the ML-PMHT plot. While this switching is not between targets with similar dynamics (as happens with Sce-

narios 2 and 5), it is happening with targets that are moving very slowly. As as result, there are no dynamics — either evolving position over time or Doppler — to differentiate measurements between two targets.

All the simulations up to this point were performed using the ML-PDA target measurement generation model — that is, at most one measurement was generated by the target in any given scan. In this condition, ML-PMHT and ML-PDA had identical performance in the single-target cases, and ML-PMHT outperformed ML-PDA in the small-separation multitarget cases. We now round out the comparison between the two algorithms by considering what happens when the ML-PMHT target measurement generation model is used to generate the data and more than one measurement is allowed to originate from the target.

In order to do this, Scenario 1 (with K-distributed clutter) was re-run for one, two and three expected target returns per scan. Results for all metrics were virtually the same between the algorithms (and similar to results shown above for Scenario 1), with the exception of the number of duplicate tracks. These results are shown in Table 3.12. Here, ML-PDA, as expected, suffers from an increasing number of duplicate tracks as N, the expected number of target measurements per scan, is increased. In contrast, ML-PMHT basically has on average no duplicate tracks for any number of target measurements per scan. With this model of target-measurement generation, ML-PMHT is the superior algorithm.

Table 3.6: In-track percentage results

	ML-PDA		ML-PMHT	
	mean	confidence interval (95 percent)	mean	confidence interval (95 percent)
Scenario 1	95.4	[94.8, 96.0]	95.4	[94.8, 96.0]
Scenario 2 Tgt 1	12.0	[7.5, 16.6]	83.9	[79.6, 88.2]
Scenario 2 Tgt 2	100	[100, 100]	87.5	[83.7, 91.3]
Scenario 2 Tgt 3	1.9	[0.0, 3.8]	53.1	[46.5, 59.7]
Scenario 3 Tgt 1	82.7	[81.0, 84.4]	82.5	[81.0, 84.0]
Scenario 3 Tgt 2	76.1	[74.2, 78.0]	77.1	[75.4, 78.9]
Scenario 4 Tgt 7	61.0	[54.7, 67.4]	91.1	[89.3, 92.8]
Scenario 4 Tgt 12	52.5	[45.9, 59.2]	96.3	[95.9, 96.8]
Scenario 5 Tgt 1	45.3	[38.4, 52.2]	63.1	[56.5, 69.7]
Scenario 5 Tgt 2	48.4	[41.5, 55.4]	63.7	[57.1, 70.2]
Scenario 1 Rayleigh	66.9	[66.0, 67.8]	67.4	[66.6, 68.3]

Table 3.7: RMSE results

	ML-PDA		ML-PMHT	
	mean	confidence interval (95 percent)	mean	confidence interval (95 percent)
Scenario 1	255.8	[229.7, 281.9]	259.2	[233.0, 285.4]
Scenario 2 Tgt 1	313.4	[257.7, 369.0]	281.4	[267.7, 295.1]
Scenario 2 Tgt 2	247.0	[241.9, 252.1]	280.5	[269.6, 291.5]
Scenario 2 Tgt 3	228.4	[186.7, 270.0]	212.4	[202.3, 222.4]
Scenario 3 Tgt 1	1227.3	[1190.2, 1264.3]	946.6	[904.4, 988.8]
Scenario 3 Tgt 2	799.7	[754.6, 844.8]	878.0	[836.1, 920.0]
Scenario 4 Tgt 7	391.3	[303.8, 478.8]	321.6	[274.5, 377.7]
Scenario 4 Tgt 12	151.6	[124.0, 179.2]	213.2	[177.2, 249.3]
Scenario 5 Tgt 1	213.1	[173.3, 252.8]	244.7	[207.1, 282.2]
Scenario 5 Tgt 2	272.5	[211.3, 333.7]	245.6	[215.2, 276.1]
Scenario 1 Rayleigh	217.6	[191.7, 243.6]	182.4	[166.4, 198.9]

Table 3.8: Fragmentation results

	ML-PDA		ML-PMHT	
	mean	confidence interval (95 percent)	mean	confidence interval (95 percent)
Scenario 1	0.01	[0.00, 0.02]	0.01	[0.00, 0.02]
Scenario 2 Tgt 1	0.00	[0.00, 0.00]	0.00	[0.00, 0.00]
Scenario 2 Tgt 2	0.00	[0.00, 0.00]	0.00	[0.00, 0.00]
Scenario 2 Tgt 3	0.00	[0.00, 0.00]	0.00	[0.00, 0.00]
Scenario 3 Tgt 1	0.03	[0.01, 0.06]	0.07	[0.03, 0.10]
Scenario 3 Tgt 2	0.24	[0.17, 0.30]	0.06	[0.02, 0.09]
Scenario 4 Tgt 7	0.00	[0.00, 0.00]	0.00	[0.00, 0.00]
Scenario 4 Tgt 12	0.00	[0.00, 0.00]	0.00	[0.00, 0.00]
Scenario 5 Tgt 1	0.15	[0.07, 0.22]	0.15	[0.09, 0.22]
Scenario 5 Tgt 2	0.09	[0.03, 0.15]	0.15	[0.08, 0.22]
Scenario 1 Rayleigh	0.07	[0.03, 0.10]	0.05	[0.02, 0.08]

Table 3.9: Duplicate track results

	ML-PDA		ML-PMHT	
	mean	confidence interval (95 percent)	mean	confidence interval (95 percent)
Scenario 1	0.18	[0.13, 0.24]	0.02	[0.00, 0.04]
Scenario 2 Tgt 1	0.00	[0.00, 0.00]	0.04	[0.01, 0.08]
Scenario 2 Tgt 2	1.07	[1.01, 1.14]	0.48	[0.39, 0.57]
Scenario 2 Tgt 3	0.00	[0.00, 0.00]	0.00	[0.00, 0.00]
Scenario 3 Tgt 1	0.03	[0.01, 0.06]	0.07	[0.03, 0.10]
Scenario 3 Tgt 2	0.24	[0.17, 0.30]	0.06	[0.02, 0.09]
Scenario 4 Tgt 7	0.00	[0.00, 0.00]	0.09	[0.05, 0.13]
Scenario 4 Tgt 12	0.09	[0.03, 0.14]	0.18	[0.12, 0.24]
Scenario 5 Tgt 1	0.04	[0.00, 0.08]	0.13	[0.07, 0.19]
Scenario 5 Tgt 2	0.02	[0.00, 0.05]	0.11	[0.05, 0.16]
Scenario 1 Rayleigh	2.10	[1.83, 2.37]	0.00	[0.00, 0.00]

Table 3.10: False track results

	ML-PDA		ML-PMHT	
	mean	confidence interval (95 percent)	mean	confidence interval (95 percent)
Scenario 1	0.07	[0.03, 0.11]	0.07	[0.03, 0.11]
Scenario 2	0.06	[0.03, 0.09]	0.15	[0.15, 0.21]
Scenario 3	0.34	[0.26, 0.42]	0.26	[0.19, 0.33]
Scenario 4	0.17	[0.10, 0.23]	0.14	[0.09, 0.20]
Scenario 5	2.13	[1.92, 2.34]	2.65	[2.44, 2.86]
Scenario 1 Rayleigh	0.08	[0.02, 0.14]	0.00	[0.00, 0.00]

Table 3.11: False track length results

	ML-PDA		ML-PMHT	
	mean	confidence interval (95 percent)	mean	confidence interval (95 percent)
Scenario 1	3.14	[2.59, 3.69]	3.14	[2.56, 3.73]
Scenario 2	2.75	[2.31, 3.19]	3.37	[2.81, 3.92]
Scenario 3	4.34	[3.18, 5.50]	4.42	[3.32, 5.62]
Scenario 4	6.75	[3.72, 9.78]	11.67	[7.81, 15.52]
Scenario 5	8.98	[7.81, 10.16]	8.55	[7.59, 9.51]
Scenario 1 Rayleigh	3.58	[2.39, 4.78]	0.00	[0.00, 0.00]

Table 3.12: Number of duplicate tracks as a function of N, the expected number of target measurements per scan

	ML-PDA		ML-PMHT	
N	mean	confidence interval (95 percent)	mean	confidence interval (95 percent)
1	1.32	[1.21, 1.43]	0.03	[0.00, 0.05]
2	4.05	[3.88, 4.23]	0.04	[0.01, 0.06]
3	6.12	[5.92, 6.31]	0.09	[0.05, 0.13]

3.7 Conclusions

We developed a true multitarget implementation of ML-PMHT. In the process, we developed an expression for the ML-PMHT CRLB. We showed that ML-PMHT is a statistically efficient estimator, and that the ML-PMHT CRLB can be evaluated in real time, a valuable result in and of itself – this allows for a "complete" tracker framework that provides both a state estimate as well as a reliable estimate of the state covariance.

We then tested these developments by comparing ML-PDA and ML-PMHT with Monte Carlo testing. This testing first showed that ML-PDA and ML-PHMT are effective very low observable trackers, working down to an expected target

SNR of 4-5 dB (post signal processing). After this, the ML-PDA and the ML-PMHT tracking algorithms were applied to five different benchmark scenarios with Monte Carlo trials using a target measurement generation model of zero or one measurements originating from the target in a scan. For scenarios with a single target or multiple targets with measurements that could easily be differentiated by dynamics, the performances of ML-PDA and ML-PMHT were identical — ML-PMHT did not suffer from the fact that its measurement assignment model did not match the actual target measurement generation model. In cases with closely-spaced targets with measurements that could not be differentiated easily by dynamics, ML-PMHT outperformed ML-PDA due to the fact that the former had a true multitarget LLR formulation, while the latter had to handle multiple targets in a sequential single-target mode. Finally, when the target measurement generation model was switched to that of ML-PMHT, with multiple measurements per scan being generated by the target, ML-PMHT outperformed ML-PDA in terms of the number of duplicate tracks generated. Overall, the performance of ML-PMHT makes it the preferred algorithm.

Chapter 4

Measurement Spaces and Maneuver-Model Parameterization

4.1 Introduction

The previous chapter showed that ML-PMHT has several advantages over ML-PDA. As a result, in this chapter we continue to focus on ML-PMHT. First, we compare the performance of ML-PMHT between the "natural" measurement space for multistatic trackers and the Cartesian measurement space. We then introduce a maneuver model parameterization for the tracker that improves its performance when following sharply maneuvering targets. Finally, we extend the CRLB work started above for the straight-line parameterization to the maneuver-model parameterization. A portion of this chapter will appear in *Transactions on Aerospace and Electronics Systems* as "The ML-PMHT Multistatic Tracker for Sharply Maneuvering Targets."

We start by first examining the performance of ML-PMHT for a multistatic system

implemented in Cartesian measurement space to that of ML-PMHT implemented in delay-bearing measurement space. Many other (Kalman-based) trackers operate in Cartesian space in order to achieve a linear state transition. (Often in this space, the measurement relationship is still nonlinear and must be linearized.) Such trackers can also operate in delay-bearing space via methods such as the extended Kalman filter (EKF) or the unscented Kalman filter (UKF) — an example of this is found in [27] — but this involves approximations to both the state transition and measurement equations. In contrast, it is possible to implement ML-PMHT (or ML-PDA) without any linearizing approximations in either measurement space. In a multistatic active framework, the measurement covariance is more accurately represented in delay-bearing space, so it should be advantageous to operate in this space. We verify this, and quantify the difference, by comparing the performance of the two implementations via Monte Carlo simulation.

Next, we develop and test a maneuvering-model parameterization for ML-PMHT. As stated above, ML-PMHT must make an assumption about a target's motion; it parameterizes this motion over a batch of measurements with some vector (which includes the maneuver parameterization). In previous work, the target motion was always parameterized as a straight line, but obviously, not all targets move in straight lines. Because of this, the ML-PMHT tracker was implemented with a sliding batch/window (described in Chapter 2), with the assumption that a

maneuvering target trajectory could be reasonably approximated with a series of line segments. This allowed ML-PMHT to track moderately maneuvering targets, but not sharply maneuvering targets. To overcome this, we add a maneuver time t_m and a maneuver angle θ_m to the parameter set describing the target, which allows for a maneuver within the batch.

Finally, we extend the work above in Section 3.4.3 (also in [63]) that developed an expression for the Fisher Information Matrix (FIM) for the ML-PMHT tracker. This work covered straight-line target parameterizations; we expand this to cover the maneuvering-model parameterization developed in Section 4.3. With the FIM, we can obtain the CRLB for ML-PMHT, and we show that, as in the case of straight-line parameterization, ML-PMHT with the maneuvering-model parameterization is an efficient estimator. As a result, the CRLB can be used in an ML-PMHT tracking implementation to provide an estimate for the covariance of any output tracks.

4.2 Cartesian measurement space vs. delay-bearing measurement space

In previous work (see [36], [58], [59], and [60]), the ML-PMHT algorithm was implemented in Cartesian measurement space. Measurements of time-delay and azimuth were converted to (x, y) Cartesian coordinates using the bistatic equa-

tions of [26]. The Gaussian-distributed measurement errors in time-delay and azimuth space were converted to (approximate) Gaussian distributions in Cartesian space following the work of [25]. This approach worked reasonably well as long as the actual measurement errors were relatively small. However, as the errors (especially the azimuthal error) grow, the accuracy of the Gaussian assumption for the converted measurements starts to break down. This becomes a problem not only for ML-PMHT, but for all trackers operating in Cartesian measurement space. Fortunately, ML-PMHT can operate without any linearizing approximations in the delay-bearing measurement space. In contrast, many Kalman-filter based trackers stay in Cartesian measurement space in order to maintain at least a linear state transition.

To get a feel for the errors caused by converting measurements and their associated covariances to Cartesian space, consider the following simple example of a target at a range of 10,000 distance units (all distances in this work have arbitrary units) with a true bearing of 030° relative to north (assume that both the source and the receiver are at the origin in this case). Measurement errors are simulated for $\sigma_t = 0.1$ seconds and $\sigma_\theta = 1°$. These simulated points are plotted (after conversion into Cartesian coordinates) in Figure 4.2. The ellipse in this figure is the Gaussian-based 95% probability region. In this situation, this approximation fits the data very well. Now consider the same geometry and time error, but with an azimuthal

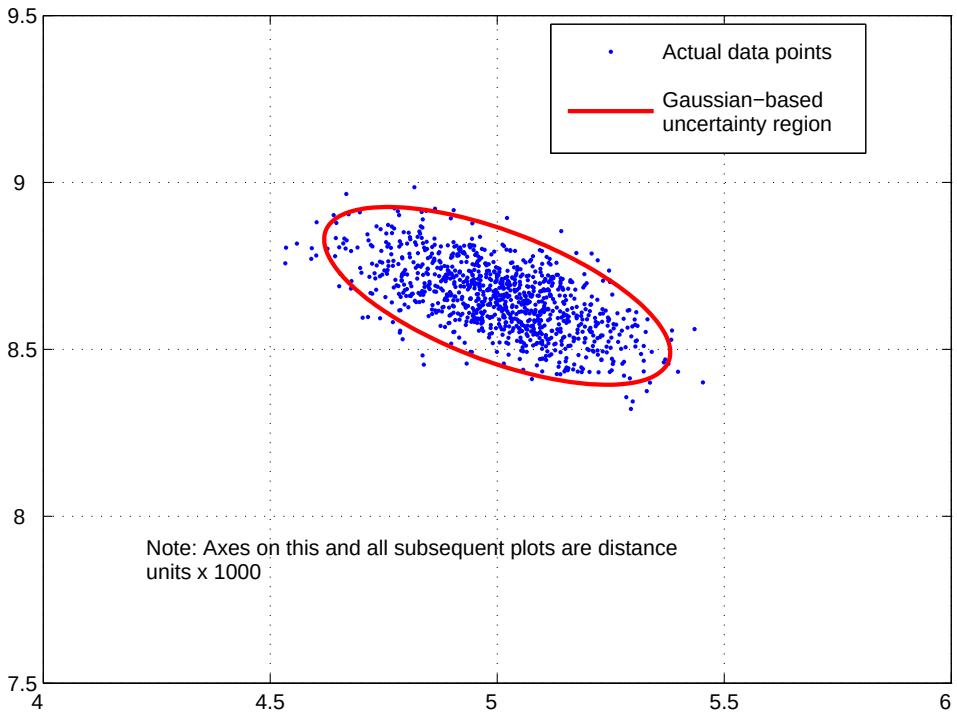

Fig. 4.1: Simulated points and equivalent Gaussian uncertainty region for $\sigma_t = 0.1$ s and $\sigma_\theta = 1^\circ$

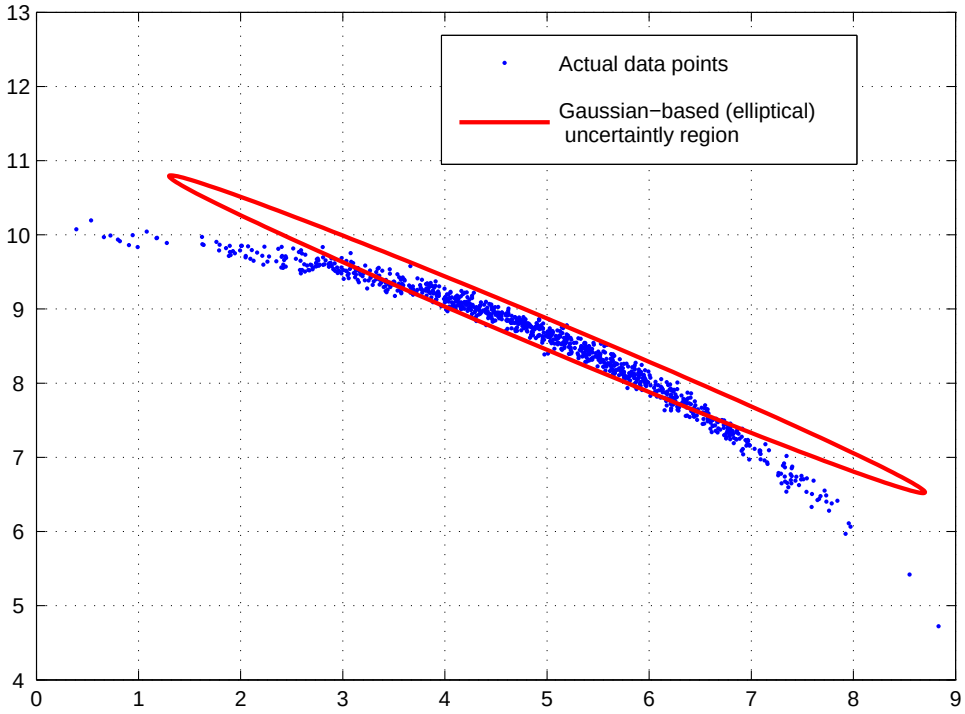

Fig. 4.2: Simulated points and equivalent Gaussian uncertainty region for $\sigma_t = 0.1$ s and $\sigma_\theta = 10°$

error of $\sigma_\theta = 10°$. Again, simulated measurements with the equivalent Gaussian-based ellipse are shown in Figure 4.2. It is apparent in this case that the Gaussian is no longer a good approximation to the actual distribution of the measurement data: the Gaussian-based ellipse cannot cover the "crescent-shaped" uncertainty region.[1]

Another way to think about this is in terms of data fusion. The ML-PMHT like-

[1] This is the so-called "contact lens problem" discussed in [72] where recursive estimators were considered. Here we use a batch estimator for low-SNR targets.

lihood ratio (5.2.1) implicitly fuses measurements that are close together. If the uncertainty regions of a given set of measurements overlap, then these measurements are effectively fused together and used in the determination of the target solution. Figure 4.3 illustrates this. Two target-originated measurements (from two different source-receiver pairs) are plotted in relation to the target. Their approximate Gaussian-based ellipses do not overlap each other (or the target location), so it is unlikely that these measurements will be associated together — the chances are good that both will be ignored in the solution determination. In contrast, the correct "crescent" uncertainty regions do overlap, so with this representation, cross-sensor data association and fusion are possible and the measurements will (correctly) be used in the determination of the target solution $\mathbf{x}$.

Consider now a target moving in a straight line and a tracker operating in delay-bearing measurement space. Here, the measurement equation is highly nonlinear. To review, the target motion (assumed to be deterministic over the relatively short time interval covered by a batch of measurements) is parameterized by the vector

$$\mathbf{x} = (x_0 \;\; \dot{x} \;\; y_0 \;\; \dot{y})^T \tag{4.2.1}$$

The Cartesian measurement matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \begin{bmatrix} 1 & t & 0 & 0 \\ 0 & 0 & 1 & t \end{bmatrix} \tag{4.2.2}$$

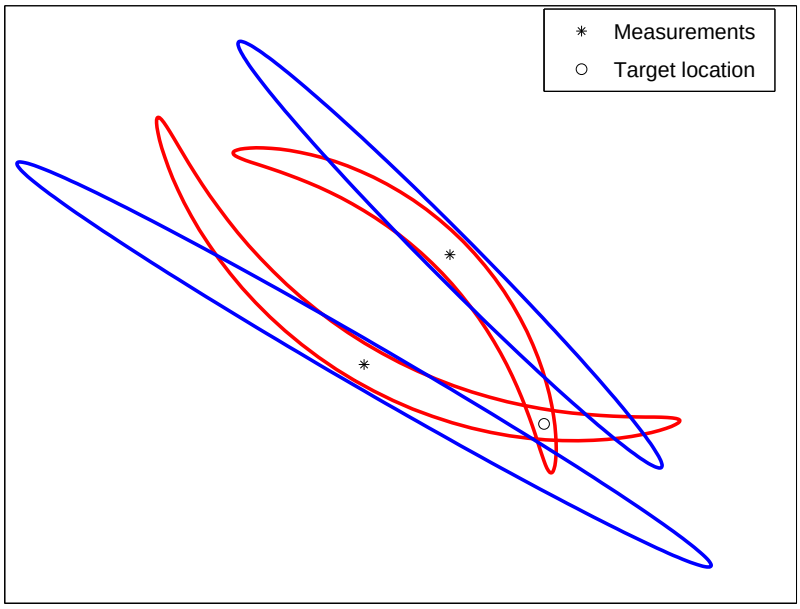

Fig. 4.3: Gaussian-approximated vs. actual (crescent) uncertainty regions. In the Gaussian approximation, target-originated measurements will be ignored.

so the predicted Cartesian measurements at time t for this state vector are given by

$$\begin{pmatrix} x \\ \\ y \end{pmatrix} = \mathbf{Hx} \tag{4.2.3}$$

We can now use these predicted Cartesian measurements to get the predicted time delay

$$\tau = \frac{r_{TS} + r_{TR}}{c} \tag{4.2.4}$$

where r_{TS} is the distance from the predicted measurement to the source, r_{TR} is the distance from the predicted measurement to the receiver, and c is the speed of sound. The azimuth is given by

$$\theta = \tan^{-1}\left(\frac{y - y_R}{x - x_R}\right) \tag{4.2.5}$$

Here, x_R and y_R are the (known) Cartesian positions of the receiver. This state-to-measurement conversion is highly nonlinear, making it difficult to use Kalman-based estimation techniques. In contrast, the ML-PMHT algorithm can operate without any linearization in Cartesian space or in delay-bearing measurement space. For a given set of measurements, it finds the parameterization of the target motion producing predicted measurements that maximize the ML-PMHT LLR (2.1.18). In delay-bearing space, these predicted measurements are produced by taking an initial target state (4.2.1) and propagating the target forward to the time of the measurement according to (4.2.3). Then, given the target's assumed

(x, y) position, the predicted delay time and azimuth measurements τ and θ are calculated via (4.2.4) and (4.2.5).

4.2.1 Data set description

A data scenario was created to compare the performance of the Cartesian vs. delay-bearing implementation of ML-PMHT. This data scenario was set up to match the first scenario in the Metron 2009 dataset [51], which was a simulated benchmark multistatic data set put out for use by the Multistatic Tracking Working Group (MSTWG). The scenario (shown in Figure 4.4) features four targets, with 25 receivers and four transmitters, set out over an approximate 60,000-by-60,000 distance unit square grid. Each target does a single revolution in a rectangular pattern. The simulation features two types of pings — a CW ping with a delay-time error of $\sigma_t = 0.1$ s, and an FM ping with $\sigma_t = 0.01$ s. On average, there are approximately 35 clutter returns per scan, and the majority of the target returns have an SNR between 4.95 dB (the detector threshold) and 9 dB. The average target probability of detection in a given scan is $P_d = 0.11$. These produce typical values of $\pi_0 \approx 0.99$ and $\pi_1 \approx 0.01$ in the LLR (2.1.19). Under such conditions the EKF or other recursive algorithms may encounter difficulties. Finally, what makes this dataset interesting from the point of view of delay-bearing processing is the azimuthal uncertainty of the receivers — this uncertainty is set at $\sigma_\theta = 8°$, which will produce an uncertainty region similar to that shown in Figure 4.2.

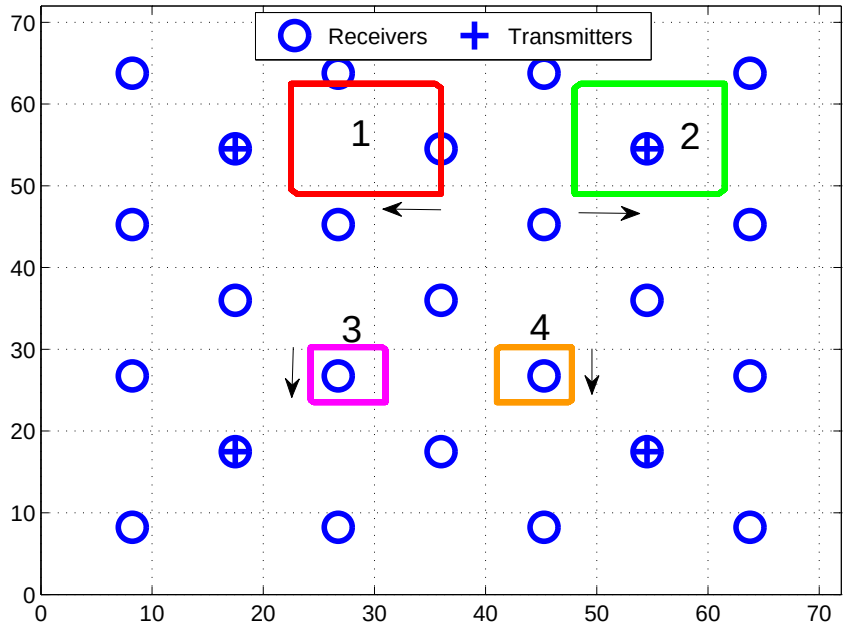

Fig. 4.4: Multistatic scenario used for Monte Carlo runs

4.2.2 Monte Carlo results

For the scenario shown in Figure 4.4, 200 runs were performed with Cartesian processing and delay-bearing processing. For each run, the following metrics were evaluated: target in-track percentage, root mean-square error (RMSE), track fragmentation, number of duplicate tracks, number of false tracks, and mean false track length. These metrics are the same as those used for the work in Chapter 3 and are all defined in section 3.5.2. All results are presented in Tables 4.1 and 4.2.

Overall, the delay-bearing implementation clearly outperformed the Cartesian im-

plementation. In particular, two metrics stand out. First, the delay-bearing implementation was nearly 20 points better in some cases in terms of in-track percentage. The delay-bearing implementation also performed much better — in all cases by a factor of almost two — in terms of track fragmentation. This is a measure of the algorithm's ability maintain a continuous (unbroken) track. The number of duplicate tracks was slightly higher for the delay-bearing implementation, but this turned out to be somewhat of a unique case that was due to how the duplicate track metric is calculated.[2] Overall, the more accurate measurement covariance for the delay-bearing implementation allows ML-PMHT to better initiate and maintain track on a target.

An example result for the scenario using the ML-PMHT Cartesian implementation is shown in Figure 4.5. A result from same scenario using the ML-PMHT delay-

[2] As is stated above, a duplicate track is counted if there is any overlap (even just one point) between two tracks on the same target. In the Cartesian case, there were large gaps between track segments. In the delay-bearing case, the track segments were on average much closer to each other (as is seen by the higher P_{DT} values). In many cases a track would go all the way to the end of a leg. It would be unable to "turn the corner" with the target, and a new track would be started at the corner on the new leg. There would be one point of overlap for the two tracks at the corner, generating a duplicate track. In contrast, for the Cartesian implementation, usually the first track would not make it all the way to the corner, or a second track would not start up right away on the second leg. Thus, the poorer P_{DT} results for the Cartesian implementation are "suppressing" the duplicate track results.

Table 4.1: Metrics from straight-line MC testing from Metron Scenario 1

Straight line				
	Cartesian		Delay-Bearing	
	Mean	Conf. Int.	Mean	Conf. Int.
Target In-Track Percentage				
Tgt 1	73.3	[70.6, 76.0]	72.5	[69.2, 75.9]
Tgt 2	58.1	[54.6, 61.8]	67.1	[64.5, 69.8]
Tgt 3	78.6	[75.6, 81.7]	99.7	[99.6, 99.9]
Tgt 4	81.3	[78.6, 84.0]	99.5	[99.2, 99.9]
RMSE				
Tgt 1	965	[919, 1010]	771	[711, 833]
Tgt 2	1216	[1154, 1278]	1127	[1090, 1165]
Tgt 3	951	[909, 992]	671	[645, 697]
Tgt 4	914	[871, 957]	786	[755, 818]
Track Fragmentation				
Tgt 1	1.70	[1.58, 1.83]	0.63	[0.55, 0.70]
Tgt 2	2.01	[1.88, 2.13]	0.48	[0.39, 0.57]
Tgt 3	1.68	[1.55, 1.80]	0.07	[0.03, 0.12]
Tgt 4	1.58	[1.45, 1.72]	0.05	[0.02, 0.08]
Number Duplicate Tracks				
Tgt 1	0.68	[0.58, 0.79]	0.87	[0.75, 0.99]
Tgt 2	0.62	[0.50, 0.75]	0.98	[0.88, 1.09]
Tgt 3	0.67	[0.56, 0.74]	1.09	[0.97, 1.20]
Tgt 4	0.65	[0.54, 0.76]	1.51	[1.40, 1.62]
Number False Tracks				
	3.84	[3.57, 4.11]	3.50	[3.33, 3.66]
Mean False Track Length				
	4.53	[4.33, 4.73]	7.02	[6.81, 7.23]

Table 4.2: Metrics from maneuver-model MC testing from Metron Scenario 1

Maneuver model						
	Cartesian		DB MMFL		DB MMVL	
	Mean	Conf. Int.	Mean	Conf. Int.	Mean	Conf. Int.
Target In-Track Percentage						
Tgt 1	76.0	[73.0, 79.0]	99.1	[98.3, 100]	100.0	[100, 100]
Tgt 2	68.1	[64.5, 71.7]	95.1	[93.0, 97.1]	99.96	[99.91, 100]
Tgt 3	82.7	[80.3, 85.2]	99.9	[99.8, 100]	99.86	[99.75, 100]
Tgt 4	82.7	[79.7, 85.6]	99.9	[99.8, 100]	99.95	[99.88, 100]
RMSE						
Tgt 1	872	[826, 917]	388	[347, 429]	193	[187, 199]
Tgt 2	1000	[947, 1053]	617	[562, 673]	226	[218, 233]
Tgt 3	854	[810, 899]	317	[301, 334]	207	[200, 213]
Tgt 4	862	[816, 909]	404	[380, 429]	219	[211, 226]
Track Fragmentation						
Tgt 1	1.44	[1.30, 1.58]	0.03	[0.01, 0.05]	0	[0, 0]
Tgt 2	1.66	[1.53, 1.80]	0.12	[0.08, 0.17]	0.01	[0.00, 0.02]
Tgt 3	1.37	[1.24, 1.50]	0	[0, 0]	0	[0, 0]
Tgt 4	1.42	[1.29, 1.54]	0.03	[0.00, 0.05]	0	[0, 0]
Number Duplicate Tracks						
Tgt 1	0.14	[0.09, 0.20]	0.27	[0.20, 0.34]	0	[0, 0]
Tgt 2	0.15	[0.09, 0.21]	0.45	[0.38, 0.52]	0	[0, 0]
Tgt 3	0.09	[0.05, 0.13]	0.09	[0.05, 0.13]	0.01	[0, 0.01]
Tgt 4	0.20	[0.13, 0.25]	0.25	[0.19, 0.31]	0.01	[0, 0.01]
Number False Tracks						
	1.92	[1.72, 2.11]	0.28	[0.19, 0.31]	0	[0, 0]
Mean False Track Length						
	4.67	[4.24, 5.11]	2.03	[1.41, 2.66]	0	[0, 0]

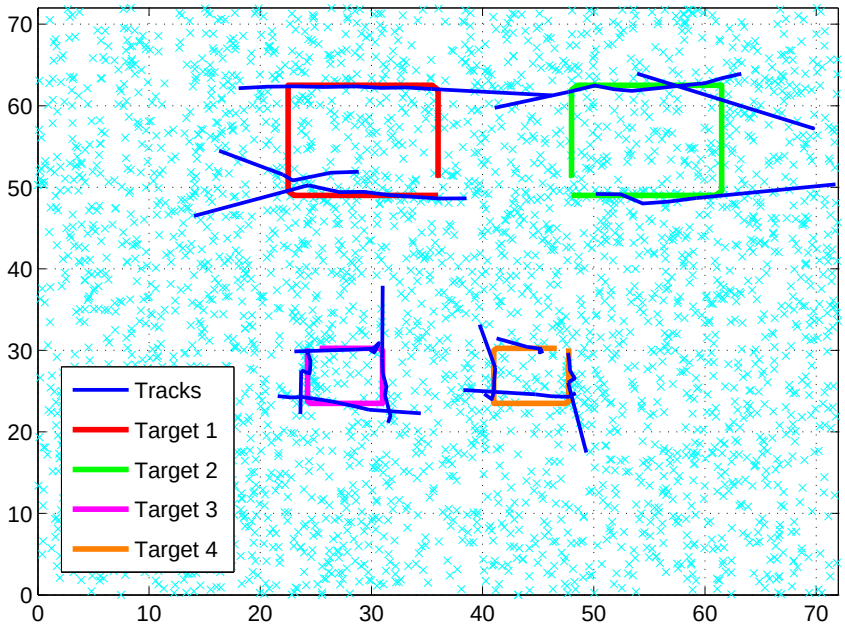

Fig. 4.5: Example plot with Cartesian straight-line parameterization. Measurements from first batch are shown.

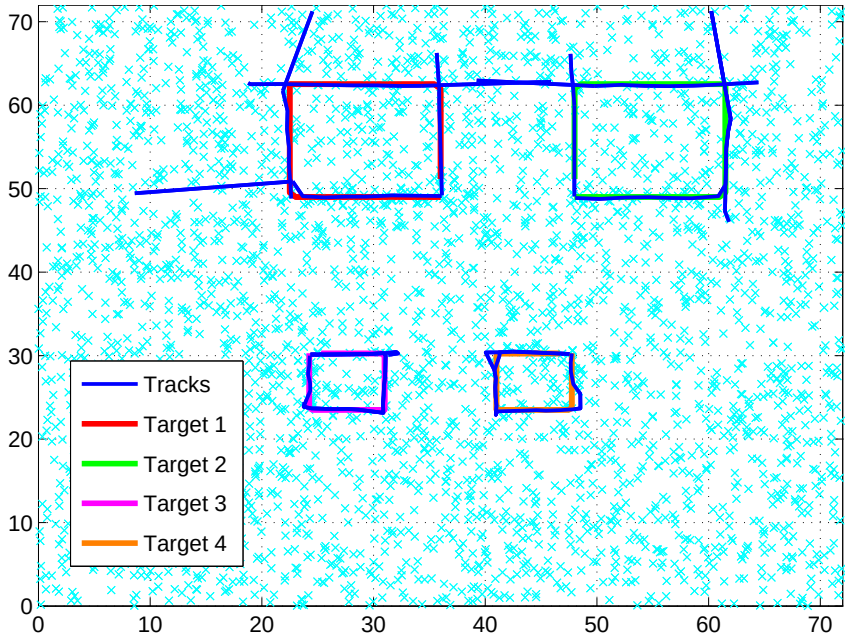

Fig. 4.6: Example plot with delay-bearing straight-line parameterization. Measurements from first batch are shown.

bearing implementation is shown in Figure 4.6. There is a noticeable difference between these results — the delay-bearing processing shows cleaner, more accurate tracks, with higher in-track percentage, which is consistent with the results seen in Tables 4.1 and 4.2.

4.3 Maneuvering-model parameterization

We now introduce a maneuvering-model parameterization that allows ML-PMHT to track sharply maneuvering targets. Up to this point, the ML-PMHT tracker

was implemented in a sliding-batch manner, and within a single batch, the target motion was assumed to be a straight line that could be completely described by the parameter vector in (4.2.1). As long as the target was not maneuvering severely, this sliding batch, straight-line parameterization was sufficient to track through maneuvers, as is illustrated with a simple example in Figure 4.7.[3] (This figure is the same as Figure 2.1; it is reproduced here for convenience.) However, when the target maneuvers more severely, the straight-line parameterization causes ML-PMHT to track off the target, which is shown with another simple example in Figure 4.8. This can also be seen in actual tracking examples in the previous section, in Figures 4.5 and 4.6. In these examples, for some of the targets ML-PMHT has good a track initially; then the target maneuvers suddenly by 90°. The tracker is not able to follow the target; the best straight-line solution it can come up with misses the maneuver, and the track ends up being dead-reckoned off on the original path until it is dropped. (New tracks are initialized after the maneuvers, but this is still at best a break in the track.) In order to enable ML-PMHT to track targets performing these more severe maneuvers, we now add two components to the parameter vector, a maneuver time t_m and a maneuver angle

[3] The actual tracker implementation is similar to that described in Chapter 2, with some slight modifications, in order to match the Metron dataset parameters. In this case, each batch is 1800 seconds (10 time updates) long. At the end of each tracker update, the batch is slid forward by 360 seconds.

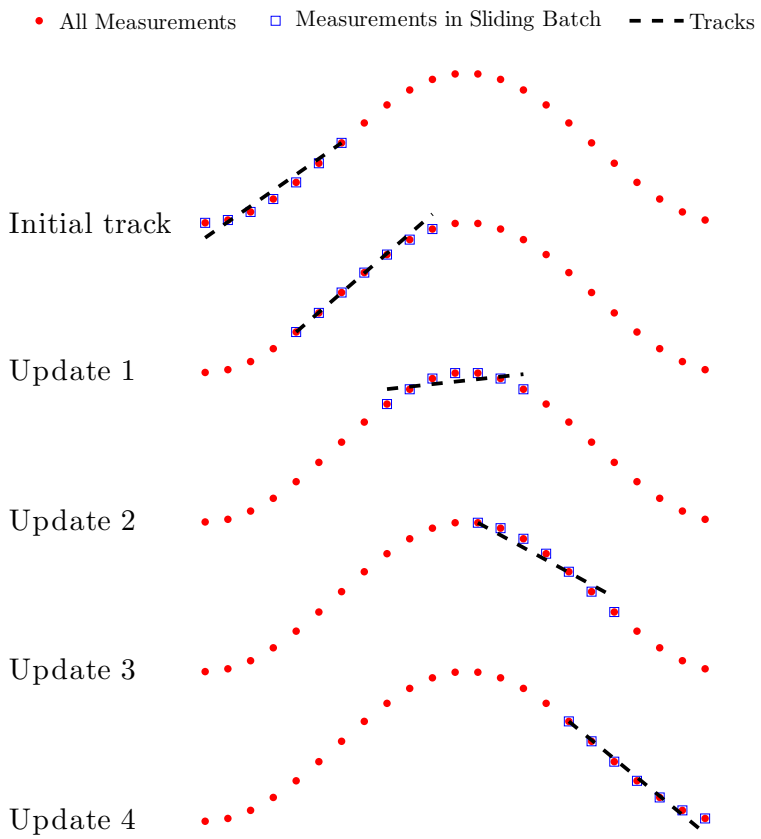

Fig. 4.7: Sliding window implementation for ML-PMHT in the case of a moderately-maneuvering target. For the measurements in each sliding batch (denoted by blue squares), ML-PMHT straight-line solutions are shown.

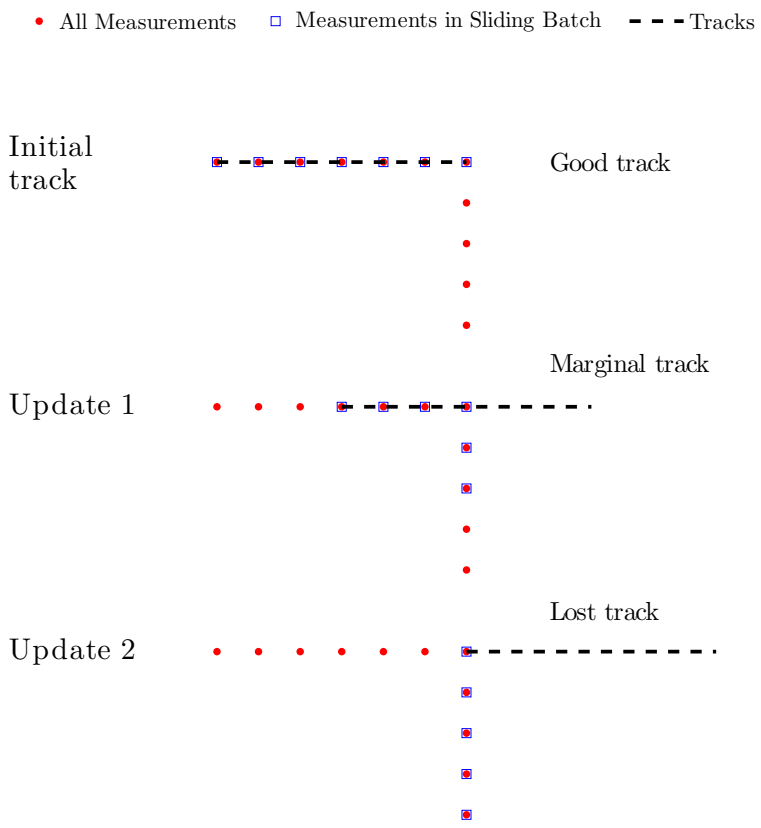

Fig. 4.8: Sliding window straight-line parameterization for ML-PMHT cannot follow sharply maneuvering target. Track is lost on the target.

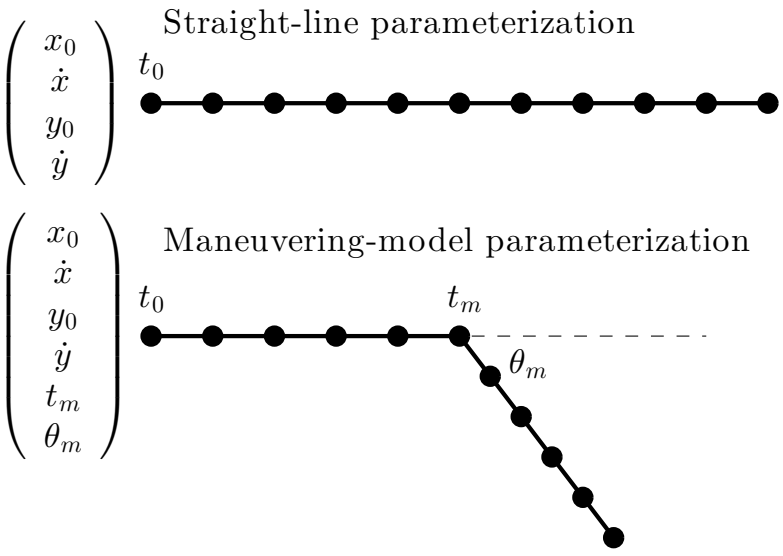

Fig. 4.9: Straight-line parameterization versus maneuvering-model parameterization

(course change) θ_m. Now, the target motion in a batch can be represented as two line segments – this new parameterization is shown in comparison to the previous straight-line parameterization in Figure 4.9.

With this maneuvering model, obtaining the predicted location of a measurement at any time during the batch is only slightly more difficult. If the time of the predicted measurement is prior to the (assumed) maneuver time, then nothing changes — the position of this predicted measurement is that given by (4.2.3) (or (4.2.4) and (4.2.5) if we are operating in delay-bearing mode). If the predicted

measurement is after the maneuver time, its location is calculated by first defining a new 4-component state $\mathbf{x}_m$ directly after the maneuver has occurred

$$\mathbf{x}_m = \mathbf{H}_m \mathbf{x} \tag{4.3.1}$$

where $\mathbf{H}_m$ is given by

$$\mathbf{H}_m = \begin{bmatrix} 1 & t_m & 0 & 0 \\ 0 & R_{00} & 0 & R_{01} \\ 0 & 0 & 1 & t_m \\ 0 & R_{10} & 0 & R_{11} \end{bmatrix} \tag{4.3.2}$$

Here, we have introduced the notation of R_{ij} being the $(i,j)^{th}$ component of a standard 2-by-2 rotation matrix with θ_m as the rotation angle. Now, to get the projected Cartesian post-measurement point, we simply project forward from the maneuver point with the (rotated) velocity vector

$$\begin{pmatrix} x_{pm} \\ y_{pm} \end{pmatrix} = \begin{pmatrix} 1 & t - t_m & 0 & 0 \\ 0 & 0 & 1 & t - t_m \end{pmatrix} \mathbf{x}_m \tag{4.3.3}$$

We combine equations (4.3.1) – (4.3.3) to obtain a final expression for the projection point at any time after the maneuver:

$$\begin{pmatrix} x_{pm} \\ y_{pm} \end{pmatrix} = \overline{\mathbf{H}} \mathbf{x}_0 \tag{4.3.4}$$

where

$$\overline{\mathbf{H}} = \mathbf{H} \qquad t \le t_m$$

$$\overline{\mathbf{H}} = \begin{bmatrix} 1 & S_{00} & 0 & T_{01} \\ 0 & T_{10} & 1 & S_{11} \end{bmatrix} \qquad t > t_m \tag{4.3.5}$$

Here, we have introduced the notation of $S_{ij} = t_m + (t - t_m)R_{ij}$ and $T_{ij} = (t - t_m)R_{ij}$. Again, if we are operating in delay-bearing space, we simply take the results of (4.3.5) and apply them to equations (4.2.4) and (4.2.5).

4.3.1 Implementing the maneuvering model

The overall logic for implementing the maneuvering-model parameterization is shown in Figure 4.10. The idea is to cycle through each existing track and check the track's maneuver status. If a maneuver is in progress, we optimize over the 4-dimensional parameter vector (4.2.1), holding the maneuver time and the maneuver angle constant. If no maneuver is in progress, we check for a new maneuver by optimizing over the parameters in (4.2.1) as well as the maneuver time and maneuver angle. If we find a maneuver time and angle that produce a better solution (in terms of a larger LLR value) than the straight-line parameterization (subject to some checks described below), we declare a new maneuver.

When checking for a new maneuver (by optimizing over the six parameters), we implement some constraints to limit the declaration of "spurious" maneuvers.

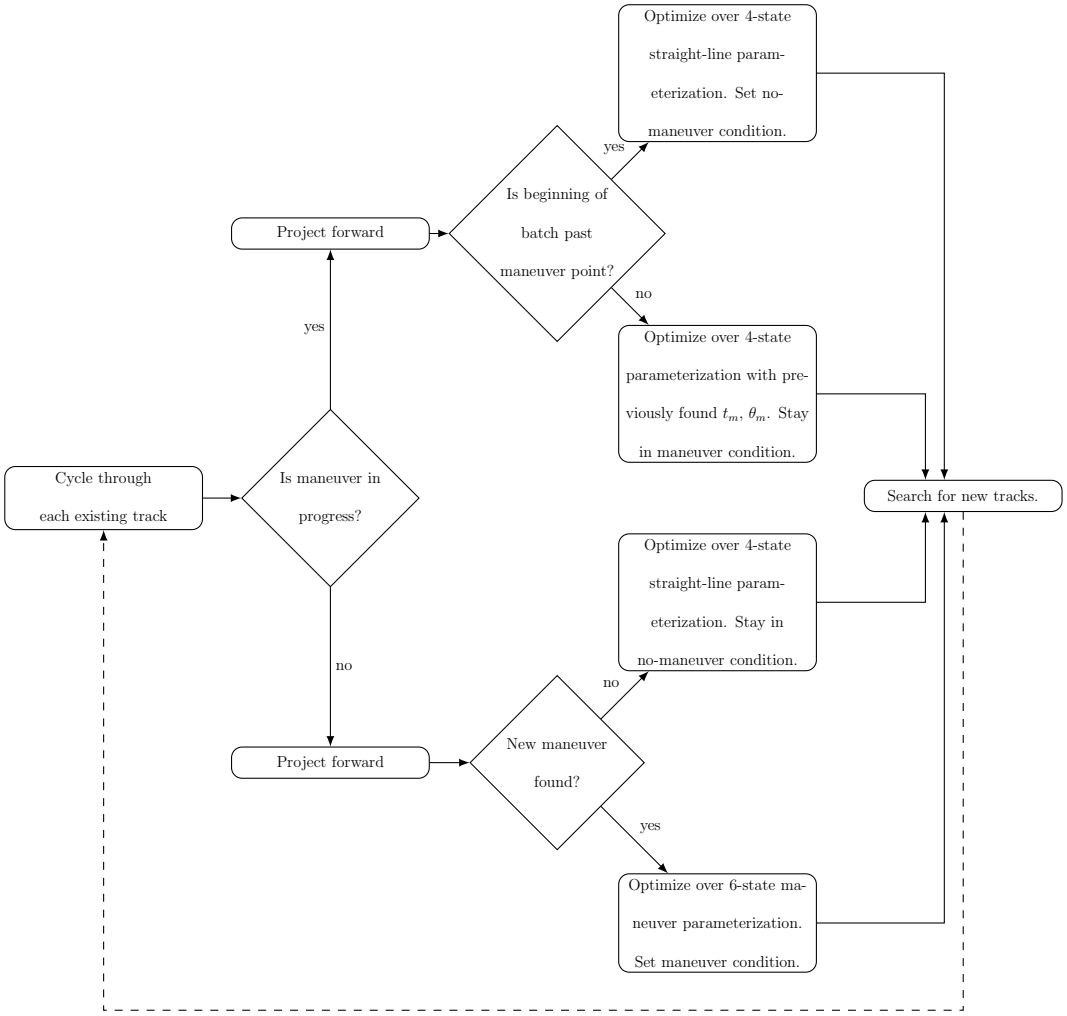

Fig. 4.10: Maneuvering-model logic diagram

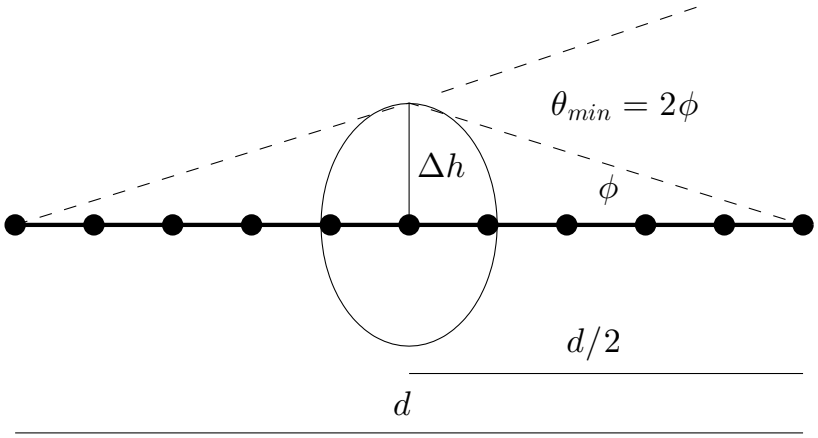

Fig. 4.11: Geometry of minimum maneuver allowed

First, we constrain the maneuver time t_m to be close to the middle of the batch. If the entire batch length is T seconds long, the only allowable maneuver times fall between $5T/12$ and $2T/3$ seconds. (A series of different window sizes was tried during development; this time window was found empirically to work the best.) Additionally, only maneuvers greater than a certain angle are allowed. Consider Figure 4.11 — this shows an example of a "maneuver" that is just caused by measurement noise; we want to avoid declaring this a maneuver. Let d be the distance a target moves in a batch of length T, and Δh be the "cross-range"

component of a "typical" measurement covariance. For this typical measurement covariance, we select a geometry where the range from target to receiver (r_{TR}) is small, giving a high probability of target detection. If σ_θ is the receiver's angular measurement uncertainty, then to a good approximation

$$\Delta h \approx r_{TR}\sigma_\theta \tag{4.3.6}$$

From the geometry of Figure 4.11, it easy to see that

$$\theta_{min} = 2\tan^{-1}\left(\frac{2\Delta h}{d}\right) \tag{4.3.7}$$

For example, for a value of $d = 2000$ units (a typical value for a slow speed target during a batch of data) and $r_{TS} = 2000$ units with $\sigma_\theta = 8°$, we end up with a value of $\theta_{min} = 30°$. (Previous work [63] has shown that the original straight-line parameterization can handle gradually maneuvering targets with a maneuver angle less than this value.)

The maneuvering-model parameterization was implemented in two different ways. First, it was done with a fixed-length batch size (in this work the length was ten sampling times), regardless of whether the target was in straight-line mode or in maneuvering mode. The maneuvering-model parameterization was also implemented with a variable-length batch, following the work of [23]. In this implementation, when searching for an initial maneuver, the back end of the batch was fixed, and the front end of the batch was allowed to expand out to a maximum

of twice the initial batch size. For each different batch size, a search was performed for the maneuver parameters t_m and θ_m — this is shown in Figure 4.12. The maneuver parameters that produced the largest ML-PMHT LLR (along with their corresponding batch size) were selected as a potential maneuver. If the LLR produced by this maneuver was larger than the LLR from the simple straight-line parameterization, a maneuver was declared. At this point, the maneuver parameters and the front end of the batch were fixed, while the back of the batch was advanced at each time update. When the back of the batch passed the maneuver point, the maneuver was declared over, and the process was started again.

The idea behind this variable-batch length implementation was to avoid finding maneuvers "too early." With the fixed-length batch size implementation, if a maneuver was declared with the minimum allowed amount of measurements after the maneuver time t_m, this could cause the maneuver angle to be inaccurately determined. Since the maneuver angle is fixed once a maneuver is declared, this would often lead to ML-PMHT losing track on subsequent updates — an example of this is shown in Figure 4.13.

For Cartesian processing, only the fixed-batch length method (from now on denoted MMFL) was used. For delay-bearing processing, both the fixed-batch length and variable-batch length (from now on denoted MMVL) were used.

Overall, there is a tradeoff when using the maneuver-model parameterization

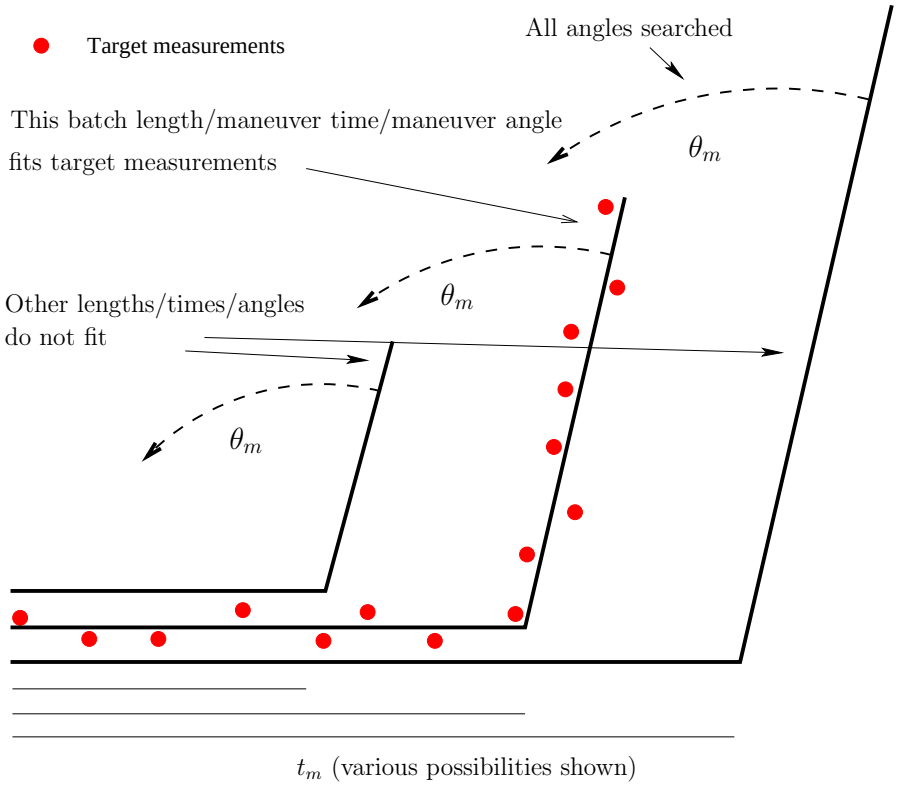

Fig. 4.12: Illustration of maneuvering-model parameterization with variable batch-length

(both MMFL and MMVL) between finding sharp maneuvers and tracking straight-line targets. The maneuver-model parameterization can be thought of as opening the bandwidth of the tracker, which makes it possible to find sharp maneuvers within a batch, but it also allows for the possibility of overparameterizing the target motion by fitting a maneuver to the measurement noise. This was seen in the preliminary work done with the maneuver-model parameterization where there was no constraint on the maneuver angles that could be found. This resulted in the tracker almost constantly fitting small maneuvers to straight-line motion — it was overparameterizing the motion. This was fixed by recognizing that the straight-line parameterization already had acceptable performance for moderately-maneuvering targets and thus it was possible to limit "found" maneuvers to those with significant maneuver angles.

We also note that (as is shown in Figure 4.10) that the maneuver model logic is only applied to *existing* tracks. Continuing with the "bandwidth" notion, a wider passband can permit more clutter to affect the solution, which in turn can adversely affect ML-PMHT's ability to find dim targets. As a result, the search for new tracks (which is done at every update) is only done with the straight-line parameterization.

Finally, for simplicity and consistency, we intentionally make it difficult to declare a maneuver, and once a maneuver is declared, the maneuver parameters (time and

angle) are held constant until the target has passed through the maneuver point. By doing this, we are again limiting the degree to which the bandwidth of the tracker is opened — this will limit the fitting of maneuvers to noise. (For the same reason we chose to only look for one maneuver at a time.) Additionally (and not insignificantly) there is also the consideration of computation time. The process of searching for the maneuver parameters increases the required computation time. If the straight-line parameterization processing time is taken as the baseline, the MMFL approach requires an increase of approximately twice the baseline computation time, and the MMVL approach requires an increase between four and five times the baseline. This increase in processing time is almost entirely due to searching for maneuver parameters when a maneuver has not been found. Not fixing the maneuver parameters (i.e. searching for the maneuver parameters at every update) would greatly increase the required processing time.

The approach we have taken is a suitable compromise between opening up the bandwidth of the tracker to find maneuvers while maintaining its original straight-line performance.

4.3.2 Maneuvering model results

The introduction of the maneuvering-model processing greatly improved the performance of the ML-PMHT tracker. For the Cartesian implementation (using the measurements converted into Cartesian coordinates), when going from the

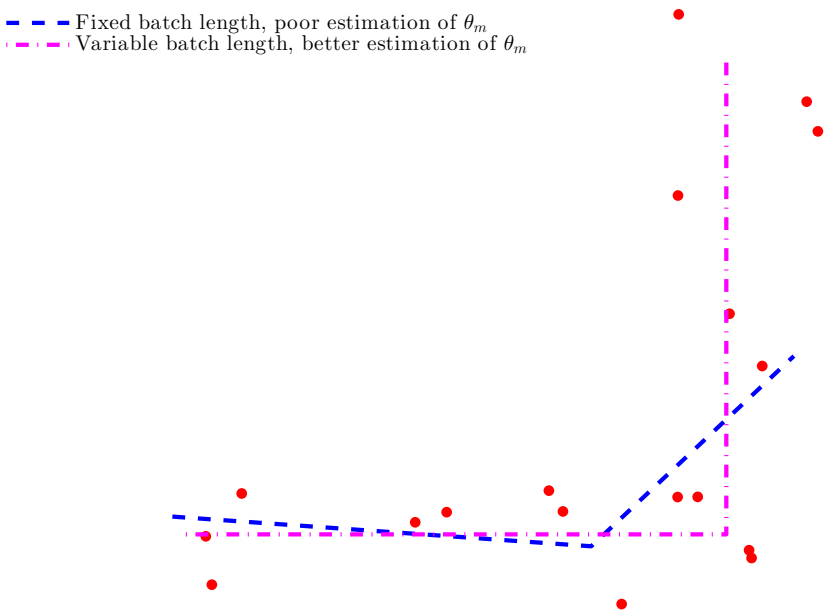

Fig. 4.13: Comparison of fixed-length batch (MMFL) vs. variable-length batch (MMVL) for maneuvering-model parameter determination. The MMFL implementation calls the maneuver early and inaccurately, whereas the MMVL implementation accurately finds the maneuver.

straight-line parameterization to the maneuvering-model parameterization, the average percentage time-in-track went up slightly, between 3% and 10%, for all four targets. For the delay-bearing implementation (i.e., using the measurements in the sensor coordinates), there were more impressive performance gains; for targets 3 and 4 with MMFL, there was increase in percentage time-in-track of nearly 30 points. For this maneuvering-model implementation, all four targets were being tracked almost 100 percent of the time. The MMVL parameterization did even better, tracking all targets in the scenario 100% of the time, with higher accuracy than the MMFL parameterization; the RMSE values for MMVL are approximately half that of MMFL.

The maneuvering-model processing also produced a slight decrease in the number of false tracks, which is somewhat counterintuitive, since the bandwidth of the tracker is being widened. This decrease in false tracks is due to the fact that with straight-line processing, sometimes a track initialized on a true target near a maneuver ended up wandering off immediately and being counted as a false track. This "wandering" was less likely with the maneuvering-model processing.

Three example plots are shown to demonstrate the progression from the Cartesian implementation of the maneuver model parameterization to the delay-bearing MMFL to the delay-bearing MMVL. There is clearly an increase in track quality from plot to plot. It is also instructive to consider the improvement in perfor-

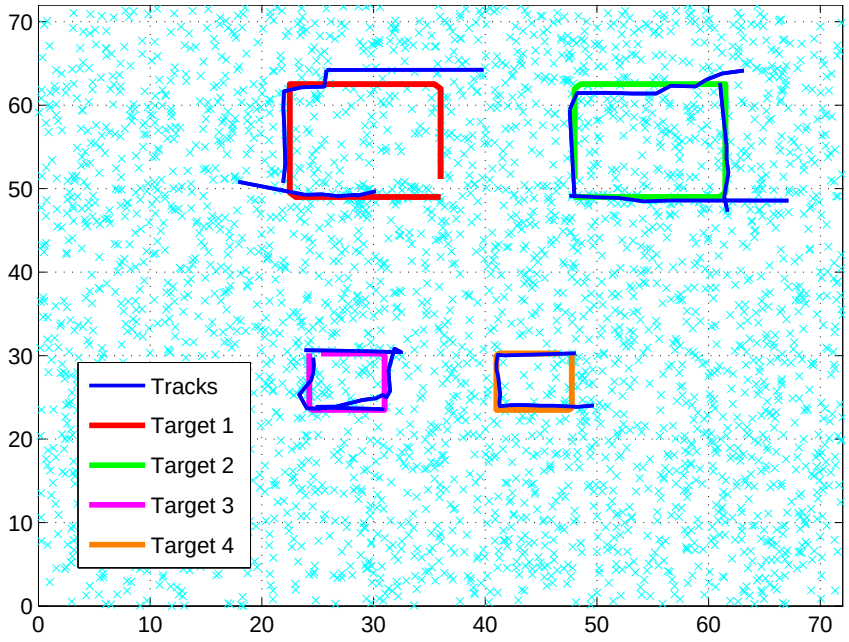

Fig. 4.14: Example of Cartesian maneuvering-model processing. Measurements from first batch are shown.

mance between the original, straight-line only Cartesian implementation to the best delay-bearing maneuvering-model version, MMVL. The time-in-track percentages for targets 1, 2, 3 and 4 were 73%, 58%, 79%, and 81% for the former, while the value was practically 100% for all four targets for the latter. Root-mean-square error decreased from approximately 1000 distance units to 200 distance units, and all other metrics showed great improvement as well. Overall, the maneuvering-model parameterization in conjunction with delay-bearing processing adds significant performance improvement to the ML-PMHT tracker.

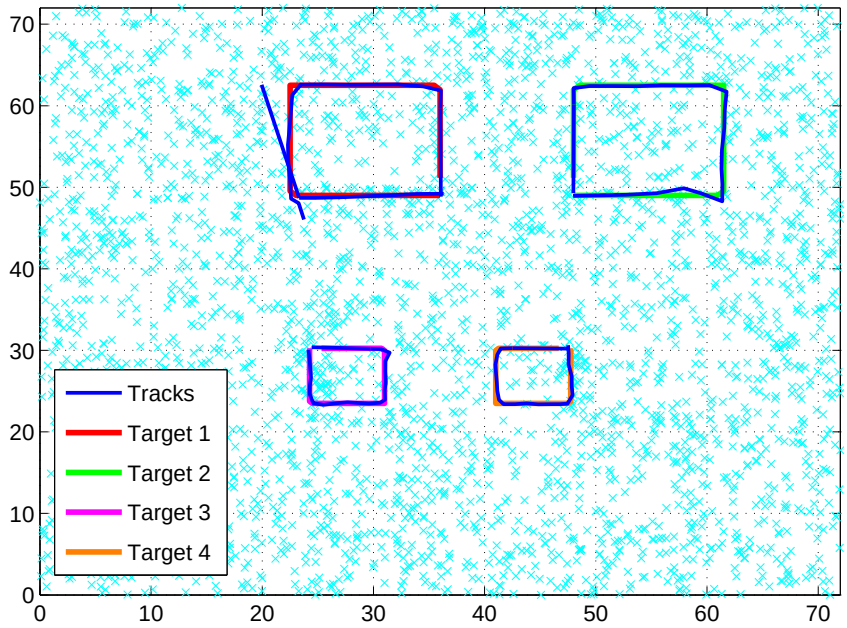

Fig. 4.15: Example of delay-bearing MMFL processing. Measurements from first batch are shown.

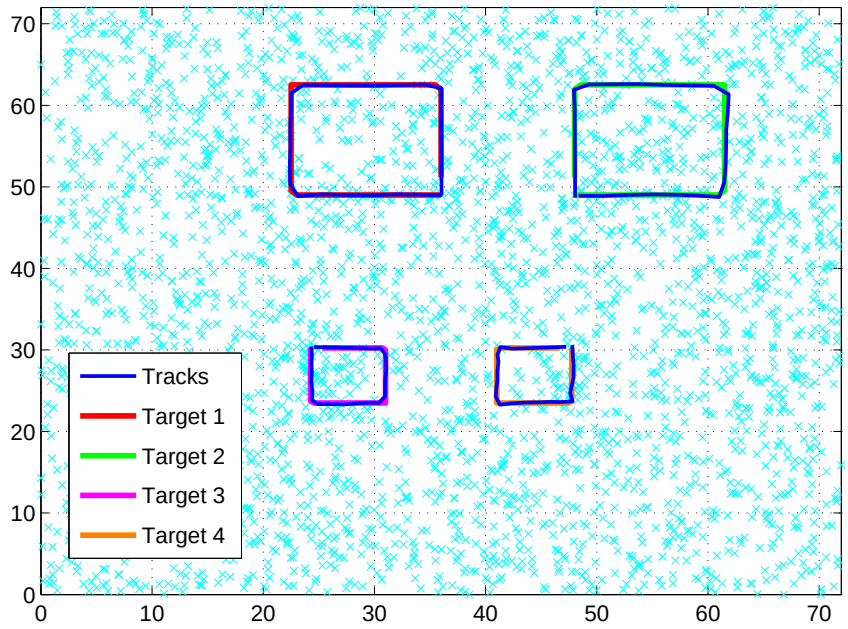

Fig. 4.16: Example of delay-bearing MMVL processing. Measurements from first batch are shown.

4.4 Maneuver model Cramér-Rao lower bound

In this section, we develop the Fisher Information Matrix (FIM) for the maneuvering-model parameterization and then check the ML-PMHT tracker for efficiency. In Section 4.4 (as well as in [63]), the FIM was developed for the ML-PMHT tracker with a straight-line parameterization; we expand on this work.

We note that the bound we are computing here is actually the *model* CRLB —

that is, the expression calculated is the lower bound given that the ML-PMHT target measurement assignment model is true.

To review, the expression for the FIM for a batch of data (for either straight-line or maneuvering-model parameterizations) is given by

$$\mathbf{J} = \sum_{i=1}^{N_w} \mathbf{J}_i \tag{4.4.1}$$

and the FIM for a single scan is given by

$$\mathbf{J}_i = \mathbf{D}_\phi^T \sum_{j=1}^{m_i} \mathbf{G}_j^T \int_{\tau_a}^{\infty}\int_{V} \frac{\frac{[\pi_1 p_1^\tau(a_j)]^2}{|2\pi\mathbf{R}_j|} e^{-\xi_j^T\xi_j}\xi_j\xi_j^T}{\frac{\pi_0 p_0^\tau(a_j)}{V} + \frac{\pi_1 p_1^\tau(a_j)}{\sqrt{|2\pi\mathbf{R}_j|}} e^{-\frac{1}{2}\xi_j^T\xi_j}} d\xi_j da_j \frac{\mathbf{G}_j}{|\mathbf{G}_j|}\mathbf{D}_\phi \tag{4.4.2}$$

(The full derivation of this expression is in Appendix A.) Here, $\mathbf{R}_j$ is the measurement covariance for the j^{th} measurement in the i^{th} scan (the scan number is implied), and $\mathbf{G}_j$ is the Cholesky decomposition of $\mathbf{R}_j^{-1}$. Finally, τ_a is the amplitude threshold, and ξ_j is either a 2-dimensional or 3-dimensional variable of integration, depending on whether Doppler information is being processed. If the measurement covariance $\mathbf{R}$ is constant within a scan (which it is above in the delay-bearing implementation), (4.4.2) can be simplified (note the j-subscripts are dropped), and the FIM for a scan reduces to a single 2-dimensional or 3-dimensional integration:

$$\mathbf{J}_i = m_i \mathbf{D}_\phi^T \mathbf{G}^T \mathbf{K}_i \mathbf{G} \mathbf{D}_\phi \tag{4.4.3}$$

where

$$\mathbf{K}_i = \frac{1}{|\mathbf{G}|} \int_{\tau_a}^{\infty}\int_{V} \frac{\frac{[\pi_1 p_1^\tau(a)]^2}{|2\pi\mathbf{R}|} e^{-\xi^T\xi}\xi\xi^T}{\frac{\pi_0 p_0^\tau(a)}{V} + \frac{\pi_1 p_1^\tau(a)}{\sqrt{|2\pi\mathbf{R}|}} e^{-\frac{1}{2}\xi^T\xi}} d\xi da \tag{4.4.4}$$

The Jacobian is defined as

$$\mathbf{D}_\phi = \begin{pmatrix} \frac{\partial \phi_1}{\partial x_0} & \frac{\partial \phi_1}{\partial \dot{x}} & \frac{\partial \phi_1}{\partial y_0} & \frac{\partial \phi_1}{\partial \dot{y}} & \frac{\partial \phi_1}{\partial t_m} & \frac{\partial \phi_1}{\partial \theta_m} \\ \frac{\partial \phi_2}{\partial x_0} & \frac{\partial \phi_2}{\partial \dot{x}} & \frac{\partial \phi_2}{\partial y_0} & \frac{\partial \phi_2}{\partial \dot{y}} & \frac{\partial \phi_2}{\partial t_m} & \frac{\partial \phi_2}{\partial \theta_m} \\ \frac{\partial \phi_3}{\partial x_0} & \frac{\partial \phi_3}{\partial \dot{x}} & \frac{\partial \phi_3}{\partial y_0} & \frac{\partial \phi_3}{\partial \dot{y}} & \frac{\partial \phi_3}{\partial t_m} & \frac{\partial \phi_3}{\partial \theta_m} \end{pmatrix} \tag{4.4.5}$$

The entries of this Jacobian matrix (which differ between the straight-line parameterization and the maneuvering-model parameterization) are now developed. In Cartesian measurement space, the predicted measurement vector ϕ is given by

$$\phi^C = (x \quad y \quad \tilde{r})^T \tag{4.4.6}$$

First, components x, y, $\dot{x}$ and $\dot{y}$ (pre- or post-maneuver) are calculated according to (4.3.1) and (4.3.4). The bistatic range-rate equation is given by [26]. With these expressions, the matrix entries of (4.4.5) can be calculated. In Cartesian processing, these expressions are relatively straightforward, and are presented in Appendix B.

In delay-bearing measurement space, the predicted measurement vector is given by

$$\phi^{DB} = (t \quad \theta \quad \tilde{r})^T \tag{4.4.7}$$

The expressions for the matrix entries of (4.4.5) in this case are slightly more complicated; however, they can be written as functions of the Cartesian Jacobian entries. They are also presented in Appendix B.

With these expressions, we can check to see if ML-PMHT with the maneuvering-model parameterization is an efficient estimator. In Chapter 3 (as well as in [63]), it was shown that the estimation error for ML-PMHT with straight-line parameterization (for a straight-line target) meets the Cramér-Rao Lower Bound (CRLB). Now, the maneuvering-model parameterization for ML-PMHT is tested to see if it meets the maneuvering-model CRLB. To this end, 500 runs were performed on a single target making an 85° turn (close to one of the turns shown above in Figure 4.4). For each run, when a maneuver was declared, the corresponding 6-parameter Fisher Information Matrix (FIM) was calculated, and the inverse of this was taken as the CRLB. With this, the normalized estimation error squared (NEES) value was calculated

$$\text{NEES} = (\hat{\mathbf{x}} - \mathbf{x}_{truth})^T \mathbf{C}^{-1} (\hat{\mathbf{x}} - \mathbf{x}_{truth}) \tag{4.4.8}$$

where $\mathbf{C}$ is the CRLB matrix, and $\mathbf{C}^{-1} = \mathbf{J}$. Each NEES value is a realization of a chi-squared random variable with 6 degrees of freedom, so the average of all 500 NEES values should be 6. The 95-percent probability interval is [5.49, 6.51]; the actual average was 6.35, which falls within the interval. A subset of 20 runs is plotted in Figure 4.17 — here we are showing solutions overlaid on the position components of the CRLB for the initial state, the maneuver state, and the final state. These results show that ML-PMHT with the maneuvering-model parameterization is an efficient estimator.

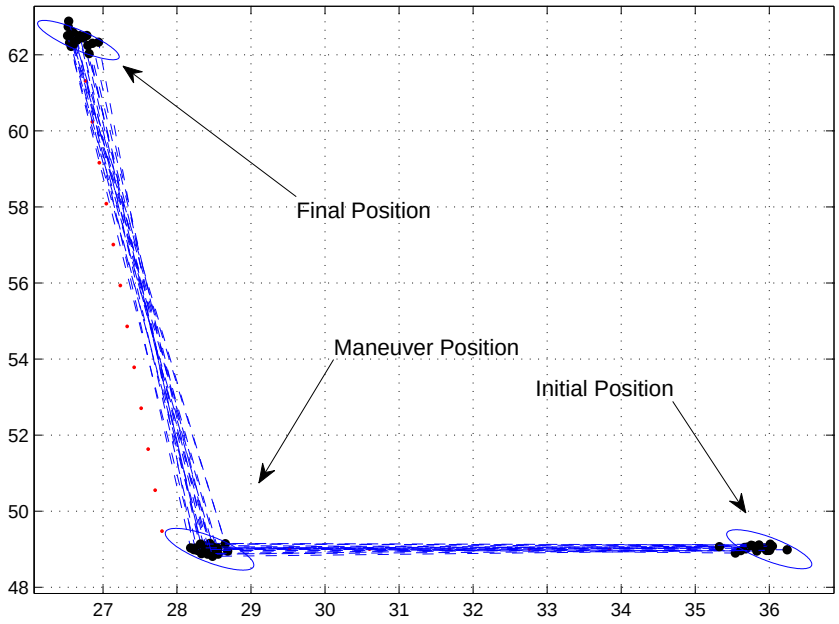

Fig. 4.17: CRLB contours for initial, maneuver, and final positions

4.5 Conclusion

The ML-PMHT tracker was first applied to a benchmark multistatic sonar simulation implemented in both Cartesian measurement space and delay-bearing measurement space. As expected, the tracker performed far better in delay-bearing space, since in this space the measurement Gaussian distribution correctly represents the uncertainty. This shows that ML-PMHT (and any other trackers that can operate in delay-bearing space) have a significant advantage when operating on multistatic sonar data over trackers that work in Cartesian measurement space. In the low-observable target-amplitude regime of the scenario, ML-PMHT

also benefited from being a batch tracker.

A maneuvering-model parameterization was developed for ML-PMHT to increase the tracker's ability to follow rapidly-maneuvering targets. This parameterization, especially when implemented in a delay-bearing variable-batch length mode, significantly increased the tracker's performance in almost all metric areas.

Finally, an expression previously developed for the ML-PMHT FIM/CRLB was extended for the maneuvering-model parameterization. Using this, the NEES calculated over Monte-Carlo trials showed that ML-PMHT with the maneuvering-model parameterization is an efficient estimator. This is a valuable result in and of itself, and it also allows the use of the CRLB as a consistent estimate for the state covariance in an ML-PMHT tracking framework.

Overall, each of the three portions of this chapter extend the capability of the ML-PMHT tracker and make the algorithm a powerful low-observable multistatic sonar tracker.

Chapter 5

Threshold Determination

5.1 Introduction

We now move onto a very different topic – we develop a novel method that can rapidly and accurately calculate tracking thresholds. Additionally, this new method, when combined with related work that will be presented in Chapter 6, will also allow for the comparison of the peaks in the LLR due to clutter and due to the target, which will yield information on whether or not a certain target is trackable in a given environment.

Previous work in [18] presented a method for determining a tracking threshold for ML-PDA; the same method can be applied to ML-PMHT. Basically, for a certain set of tracking parameters such as search volume, clutter density, probability of detection, signal-to-noise ratio, etc., the ML-PMHT LLR can be simulated and then optimized; if this is done repeatedly with Monte Carlo trials the empirical PDF (probability density function) of the maximum point in the LLR due to clutter can be calculated. The problem with this approach is that it is an extremely

time-consuming process, taking anywhere from hours to days, and if any of the parameters in the input process change, the thresholds must be re-computed. (Often, this threshold determination method takes long enough that in past works [36], [58], [60], a combination of this method and trial and error was used to find an appropriate tracking threshold for ML-PMHT implementations.) The work in [18] did also develop a "real-time" method for track-validation for ML-PDA, but its accuracy was not as good as the "off-line" approach described above. We seek the accuracy of the off-line method in real-time speed.

To do this, we take a novel approach to the problem. Instead of having a fixed set of measurements Z and finding the state parameter vector $\mathbf{x}$ that maximizes the ML-PMHT LLR for the received measurements, we assume we have some arbitrary parameter $\mathbf{x}$, and all the measurements Z are random variables that are uniformly distributed when there is no target in the search space (which as will soon be discussed, can be anywhere from one- to three-dimensional). The ML-PMHT LLR is simply treated as a transformation of the clutter (uniformly distributed) random variables. This transformation produces an expression that represents the PDF of any given point on the ML-PMHT LLR surface in the absence of a target. We then employ extreme value theory [22], [24], [28], [30], [38] to develop a PDF for the maximum value of the LLR surface. With the PDF describing the maximum value in the LLR due to clutter, we can the employ

the Neyman-Pearson lemma [49] to obtain a clutter threshold that will give us a certain false alarm ratio.

5.2 Overall framework for LLR extreme value PDF determination

In this section, we present the framework used for going from the parameters that are present in the ML-PMHT LLR to determining the extreme value distributions for the maximum points in the LLR caused by clutter measurements. The clutter PDFs will then be used in Section 5.4 to determine track acceptance thresholds.

5.2.1 Framework flow

Here, we outline the steps necessary to obtain the peak PDFs caused by clutter. First, the PDF formed by a single measurement is calculated. This single measurement PDF and the ML-PDA assumption of measurement independence are then used to derive a "batch" PDF. Next, extreme value theory is used to determine the "peak" PDF from the batch PDF. To do this, it is first necessary to calculate the number of times M the batch PDF would have to be sampled so that the maximum sample has the equivalent statistics to the maximum point that would be determined by numerically optimizing the ML-PMHT LLR. Finally, with the batch PDF and the number of samples M, the clutter peak PDF can be calculated.

Peak probability density functions are developed for four specific cases: one-

dimensional measurements (either bearings-only or time-delay-only), two-dimensional measurements (bearing and time-delay), three-dimensional measurements (bearing, time-delay, and Doppler), and two-dimensional measurements with amplitude.

5.2.2 Calculating the batch PDF via single-measurement transformation

The first step in the process is to calculate the LLR PDF from a single measurement, and then use this to obtain the LLR PDF for a batch of measurements. First, however, some background is necessary. Consider the ML-PMHT LLR, which to review, for a single target, is expressed as [77]

$$\Lambda(\mathbf{x}, Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \left\{ \pi_0 + \pi_1 V p[\mathbf{z}_j(i)|\mathbf{x}] \rho_j(i) \right\} \tag{5.2.1}$$

Here, N_w is the number of scans/frames in the batch, m_i is the number of measurements in the i^{th} scan, π_0 is the prior probability that any given measurement is from clutter, π_1 is the prior probability that any given measurement is from the target, and V is the search volume. Finally, $\rho_j(i)$ is the amplitude likelihood ratio for the i^{th} measurement in the j^{th} scan, $\mathbf{x}$ is the target parameter vector, and the PDF of the measurement, $p[\mathbf{z}_j(i)|\mathbf{x}]$, is a target-centered Gaussian.

Equation (5.2.1) is typically modified by subtracting out a constant that sets the floor of the LLR to zero. Additionally, for the time being, the amplitude is

disregarded by setting the amplitude likelihood ratio $\rho_j(i)$ to one. This leaves

$$\Lambda_{mod}(\mathbf{x}, Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \left\{ 1 + \frac{\pi_1}{\pi_0} Vp[\mathbf{z}_j(i)|\mathbf{x}] \right\} \tag{5.2.2}$$

In an ML-PMHT tracking implementation, a batch of measurements is received. Given these measurements, the ML-PMHT LLR is optimized; the state $\hat{\mathbf{x}}$ that maximizes the LLR is selected as the most likely target parameter vector.[1] If this maximum LLR exceeds a threshold, a target is declared, and $\hat{\mathbf{x}}$ defines its track; otherwise, the "detection" is rejected as being from clutter. (This approach is not unique to ML-PMHT; it is a typical maximum likelihood implementation.)

The state is assumed to have some fixed, arbitrary value (what it actually is turns out not to matter), and the measurements are treated as random variables. The measurements in this work are assumed to be from clutter (since we are looking for the probability of accepting a track when there is no target) and are thus assumed to be uniformly distributed in the measurement space [11]. The analysis starts with the LLR expression evaluated for a single measurement, which is

$$\Lambda_1(z) = \ln \left\{ 1 + \frac{\pi_1}{\pi_0} Vp(z|\mathbf{x}) \right\} \tag{5.2.3}$$

The key point is to now treat this single-measurement LLR as just a function that

[1] In this chapter the parameter vector consists of the initial state and velocity (the latter is assumed to be constant during the batch) and, consequently, it could be called "state." However, this does not apply to the maneuvering model used Chapter 3 and in [62], so we shall continue to use the term "parameter vector."

transforms a random variable; i.e. if the measurement z in (5.2.3) is a random variable (uniformly distributed since it is from clutter), then

$$w = \Lambda_1(z) \tag{5.2.4}$$

where w is just a new (transformed) random variable. In the four cases mentioned (scalar measurements, 2-D measurements, 3-D measurements, and 2-D measurements with amplitude), it is possible to get a closed-form expression for the PDF of w. Since one of the assumptions of the ML-PMHT tracker is that measurements are independent, a PDF that represents the batch ML-PMHT LLR can be obtained by either convolving the single-measurement PDF $N-1$ times (where N is the number of measurements in the batch), or by calculating the characteristic function of the single-measurement PDF, raising it to the N^{th} power, and then transforming it back to a PDF.

5.2.3 Determining the equivalent number of samples for clutter peak PDF determination

As is stated above, it is necessary to go from the batch PDF to the peak PDF. The first step is to model the effect of optimization. For an ML-PMHT tracking implementation, we would find the parameter vector $\mathbf{x}$ that yields the maximum value for the LLR with some non-linear numerical optimization scheme. Such optimization, assuming it has been implemented properly, will find the global

maximum of the LLR to within some small value δ of the true maximum value of the LLR. One conceptually simple way to obtain this optimal point would be to sample the LLR with a very finely spaced grid, consisting of M samples, and take the maximum from the sample set as the global maximum value of the LLR. Of course, this is not done in practice because M becomes prohibitively large, especially when the dimension of the state parameter vector $\mathbf{x}$ is large. However, it is possible to make some basic assumptions about the LLR surface and determine a value for the number of samples M that would be theoretically required to get the same accuracy on the global maximum as the actual numerical optimization scheme. If the batch PDF is sampled M times, and the samples are ordered, extreme value theory (discussed in the next section) can be used to calculate the PDF of the maximum sample. This PDF of the maximum value can be equated to the PDF of the peak point in the actual ML-PMHT LLR. Note that no sampling is actually required; all that is needed is the number of sampling points necessary to get the required accuracy. However, we first need a way to obtain M, and this is done with an analysis of the ML-PMHT LLR surface.

This basic sampling idea is illustrated in Figure 5.1 — in order to get within δ of the true maximum value of the LLR, we have to have at most a spacing of D (or $2\sigma_\mu$ since $D = 2\sigma_\mu$) between points.[2] Note that this figure is either showing a one-

[2] In the figure the samples are spaced symmetrically around the true peak because this is the worst-case condition in terms of minimizing δ. If the sample points were shifted left or right by

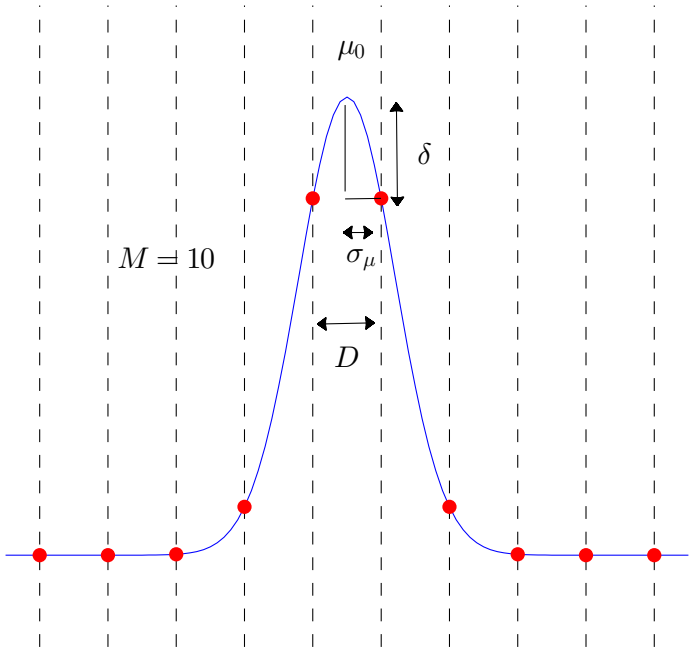

Fig. 5.1: Relationship between sampling interval and distance from LLR peak. The curve is a representative global peak in the ML-PMHT LLR.

dimensional LLR or a single-dimensional slice of a multi-dimensional LLR — the number of samples M needs to be determined for each measurement dimension. In order to obtain the relationship between δ and σ_μ, it is necessary to develop a form of the ML-PMHT LLR that is invertible — i.e. we want to know how far we have to travel away from the peak to get a certain change in LLR value. This is done by first considering the expression in (5.2.3) for the ML-PMHT LLR for a single scalar measurement. Writing this equation in a slightly different form produces

$$\Lambda_1(z) = \ln\left\{1 + K_1 e^{-\frac{1}{2\sigma_1^2}(z-\mu_0)^2}\right\} \tag{5.2.5}$$

with K_1 defined as

$$K_1 = \frac{\pi_1}{\pi_0}\frac{V}{\sqrt{2\pi\sigma_1^2}} \tag{5.2.6}$$

where σ_1 is the measurement noise standard deviation. If (5.2.5) is just treated as a function of z, and not an LLR, then by inspection, the maximum of this function occurs at $z = \mu_0$. Now, expand $\Lambda_1(z)$ with a second-order Taylor series (i.e. a parabola) in the vicinity of μ_0. This Taylor series is written as

$$\bar{\Lambda}_1(z) = \ln(1 + K_1) - \frac{1}{2\sigma_1^2}\frac{K_1}{K_1 + 1}(z - \mu_0)^2 \tag{5.2.7}$$

An actual ML-PMHT LLR calculation is performed on a batch of measurements. For N measurements (the value to use for N will be discussed shortly), the ap-

a small amount, it is obvious that the value of δ (due to whichever sample point was shifted towards the peak) would become smaller.

proximate LLR function is now written as

$$\bar{\Lambda}(Z) = \sum_{i=1}^{N} \ln(1 + K_1) - \sum_{i=1}^{N} \frac{1}{2\sigma_1^2} \frac{K_1}{K_1 + 1} (z_i - \mu_0)^2 \tag{5.2.8}$$

Note that instead of developing a Taylor series for the sum (which is what (5.2.2) is), we instead summed the Taylor series expression for each component of the sum. The former approach was avoided because this would have required a complicated multidimensional numerical integration. The latter approach gives a very reasonable result; an example is shown in Figure 5.2. This shows a sample realization of a one-dimensional clutter-only LLR, along with the Taylor series approximation given by (5.2.8).

At this point, we have a reasonable approximation for the LLR, but the measurements z_i are still present in the expression. We want a relationship between the ML-PMHT parameters and σ_μ that is not a function of the actual measurements (since we do not have the N measurements). This is achieved by replacing the measurements with the LR moments. With each z_i, a random variable, take the expected value[3] of $\bar{\Lambda}(Z)$

$$\mathrm{E}[\bar{\Lambda}(Z)] = \sum_{i=1}^{N} \ln(1 + K_1) - \sum_{i=1}^{N} \frac{1}{2\sigma_1^2} \frac{K_1}{K_1 + 1} (\mathrm{E}[z_i^2] - 2\mu_0 \mathrm{E}[z_i] + \mu_0^2) \tag{5.2.9}$$

Now, we claim that

$$E[z_i] = \mu_0 \tag{5.2.10}$$

[3] All this is essentially doing is inserting the average value of z_i and z_i^2 into (5.2.8).

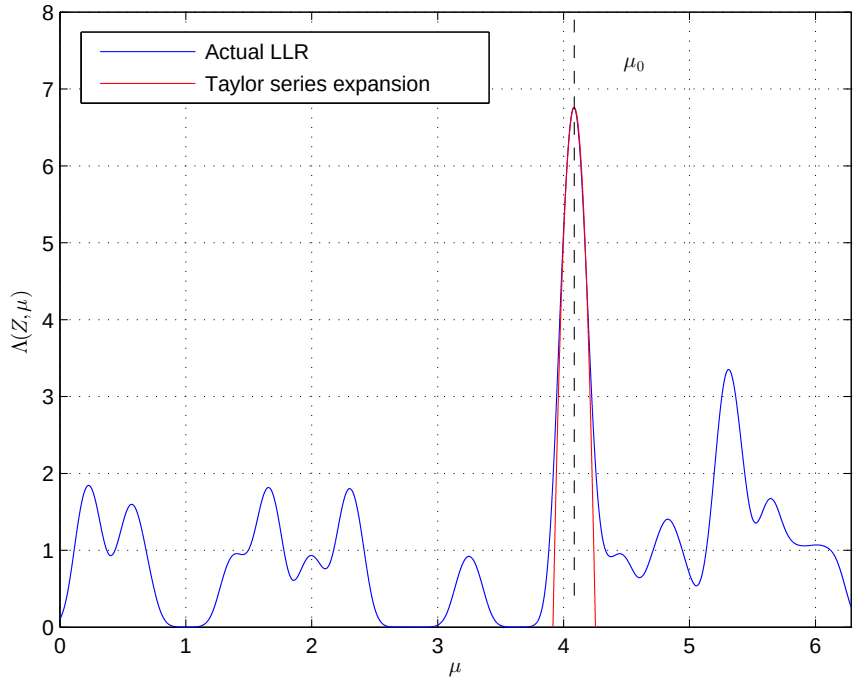

Fig. 5.2: Sample (one-dimensional) LLR, along with the Taylor series expansion in the neighborhood of μ_0 given by (5.2.8)

This is because, although the clutter measurements are uniformly distributed throughout the measurement search volume, the only measurements that affect the peak are those that are in the vicinity of the peak; it is reasonable, then, to assume the mean of these measurements is μ_0 (the location of the peak). Using this same logic, have

$$E[(z_i - \mu_0)^2] = \sigma_\mu^2 \tag{5.2.11}$$

where σ_μ is determined below. It follows then that

$$E[z_i^2] = \sigma_\mu^2 + \mu_0^2 \tag{5.2.12}$$

Finally, we recognize that the absolute value of the second term on the right-hand side of (5.2.9) is the deviation δ from the peak in Figure 5.1. Combining this term with (5.2.10) and (5.2.12) results in

$$\delta = \frac{N}{2\sigma^2}\frac{K_1}{1+K_1}\sigma_\mu^2 \tag{5.2.13}$$

Solving for σ_μ^2 produces

$$\sigma_\mu^2 = \frac{2\sigma^2}{N}\frac{1+K_1}{K_1}\delta \tag{5.2.14}$$

Now, N is really the *effective* number of measurements, i.e. the number of clutter measurements close enough the value of μ_0 that they will actually affect the main lobe, and hence the peak value. We make the assumption that only measurements within a distance of σ of μ_0 will affect this peak, giving an interval of 2σ. The number of measurements N in this interval is a discrete random variable — it is

Poisson distributed (from the ML-PMHT assumptions) with parameter ϕ, given by

$$\phi = 2\sigma\lambda \tag{5.2.15}$$

where λ is the spatial clutter density. Now, the number of samples M required to achieve an accuracy δ can be seen from Figure 5.1 to be

$$M = \frac{V}{D} + 1; \tag{5.2.16}$$

This figure also shows that $D = 2\sigma_\mu$. Combining this with (5.2.16) and (5.2.14) and taking the expectation over N (without this expectation we would be left with an M that is an RV) yields

$$M = \frac{V}{2}\sqrt{\frac{1}{2\sigma^2\delta}\frac{K_1}{K_1+1}}E[\sqrt{N}] + 1; \tag{5.2.17}$$

The expected value of $\sqrt{N}$ (which is related to the Anscombe transform [5]) is shown in Appendix C to be

$$E[\sqrt{N}] = \sqrt{\phi - \frac{1}{4}} \tag{5.2.18}$$

A question may arise at this point if it is correct to take the expected value of $\sqrt{N}$ in (5.2.17). By taking the expectation, we are essentially assuming the average value of N will generate a peak in the LLR; it would seem to make more sense that an *above-average* value of N would create a peak. However, there are two factors which make this concern irrelevant. First, M is a function of the square

root of N, a Poisson random variable, and a very interesting property of the the square root of a Poisson RV is that its variance is approximately constant – this is shown in Appendix C to be 1/4. Thus, the "spread" of M (described by its PMF) will always be known, and this spread is fairly narrow. Secondly, as will be shown shortly, the Gumbel parameters and by extension the ML-PMHT threshold change very slowly as a function of M. The rate of change of the threshold goes approximately as $\sqrt{\ln M}$. The end result of the fixed known variance of $\sqrt{N}$ and the threshold's low dependence on M is that no matter what point is used from the distribution of N, in the end the ML-PMHT threshold varies by less than a unit.

Returning to the process of obtaining a deterministic value for M, we simplify (5.2.18)

$$E[\sqrt{N}] \approx \sqrt{\phi} \tag{5.2.19}$$

Inserting this result into (5.2.17) gives the deterministic expression for M

$$M = \frac{V}{2}\sqrt{\frac{\lambda}{\sigma\delta}\frac{K_1}{K_1+1}} + 1; \tag{5.2.20}$$

Defining the spatial clutter density λ as

$$\lambda = \frac{m_{ave}}{V} \tag{5.2.21}$$

where m_{ave} is the average number of measurements in a scan, gives the final

desired expression

$$M = \frac{1}{2}\sqrt{\frac{m_{ave}V}{\sigma\delta}\frac{K_1}{K_1+1}} + 1 \tag{5.2.22}$$

This is the number of samples required for a one-dimensional measurement — for a scenario with measurements in more than one dimension, this must be calculated for each of the dimensions.

5.2.4 Extreme value theory

At this point we have the PDF for the entire ML-PMHT LLR surface caused by clutter, and the number of samples M_{tot} needed to obtain the accuracy δ that would be obtained by a properly implemented numerical optimization routine.[4] If the PDF for the entire ML-PMHT LLR surface (caused just by clutter) is sampled M_{tot} times, and the samples are ordered, the PDF of the maximum sample can be used to represent the PDF of the peak in the LLR that would be obtained through optimization.

Previous work from [18] showed that this peak PDF is a Gumbel distribution. This distribution has the form [38]

$$f(x) = \frac{1}{\beta}\exp\left[-\left(\frac{x-\nu}{\beta}\right) - \exp\left(-\frac{x-\nu}{\beta}\right)\right] \tag{5.2.23}$$

[4] Here, M_{tot} denotes the total number of measurements needed for an accuracy of δ. For multidimensional measurements, (5.2.22) needs to be applied in each dimension. Then, M_{tot} will just be the product of the M-values for each dimension.

If $F(x)$ is the CDF of the underlying distribution that is being sampled, then the parameters ν and β in (5.2.23) can be obtained with the following [22]

$$\nu = F^{-1}\left(1 - \frac{1}{M_{tot}}\right) \tag{5.2.24}$$

and

$$\beta = F^{-1}\left(1 - \frac{1}{eM_{tot}}\right) - \nu \tag{5.2.25}$$

One possible concern is that the above extreme value theory applies to a sequence of IID random variables; if we actually sample the ML-PMHT LLR at a close enough grid spacing, there will be dependence between some samples. This is because samples that are relatively close together will be influenced (i.e. the LLR will take on value) from the same measurement, so intuitively, these samples will be dependent. However, other work in extreme value theory [43] shows that the PDF of the maximum sample from an identical yet dependent sequence will asymptotically converge to the IID case. For the ML-PMHT, (5.2.22) will produce values of M_{tot} that are large enough (typically $10^7 \leq M_{tot} \leq 10^{10}$) so that it is reasonable to assume this convergence has occurred.

In order to obtain the Gumbel distribution parameters, it is necessary to obtain the CDF of the distribution that represents a batch of measurements for the ML-PMHT LLR. There is no convenient closed-form expression for this CDF that can easily be inverted. The closest available CDF will be a numerical integration of the PDF that represents the batch ML-PMHT LLR. In theory it is simple to

numerically invert the CDF of the batch and find the constants ν and β from (5.2.24) and (5.2.25). However, the number M_{tot} is typically large, so $0.99 < 1-1/M_{tot},\ 1-1/eM_{tot} < 1$. Trying to numerically invert the CDF at the extreme right side can lead to numerical calculation inaccuracies. It is more accurate to approximate the right-hand side of the CDF with an easily invertible closed-form expression. To this end, the CDF $F(x)$ is approximated with

$$F_{approx}(x) = 1 - \exp\left[-k(x-m)^l\right] \tag{5.2.26}$$

The parameters k, m, and l are fit to the numerical CDF in the region $F(x) > 0.95$ using Matlab's Optimization Toolbox. This expression is easily invertible; letting $F_{approx}(x) = \phi$ and solving for x produces

$$x = m + \left[-\frac{1}{k}\ln(1-\phi)\right]^{\frac{1}{l}} \tag{5.2.27}$$

At this point, combining equations (5.2.27), (5.2.24), and (5.2.25) gives a final expression for the parameters of the Gumbel distribution

$$\nu = m + \left[\frac{1}{k}\ln(M_{tot})\right]^{\frac{1}{l}} \tag{5.2.28}$$

and

$$\beta = \left[\frac{1}{k} + \frac{1}{k}\ln(M_{tot})\right]^{\frac{1}{l}} - \left[\frac{1}{k}\ln(M_{tot})\right]^{\frac{1}{l}} \tag{5.2.29}$$

With this, it is possible to write the Gumbel distribution that represents the peak in the ML-PMHT LLR surface caused by clutter.

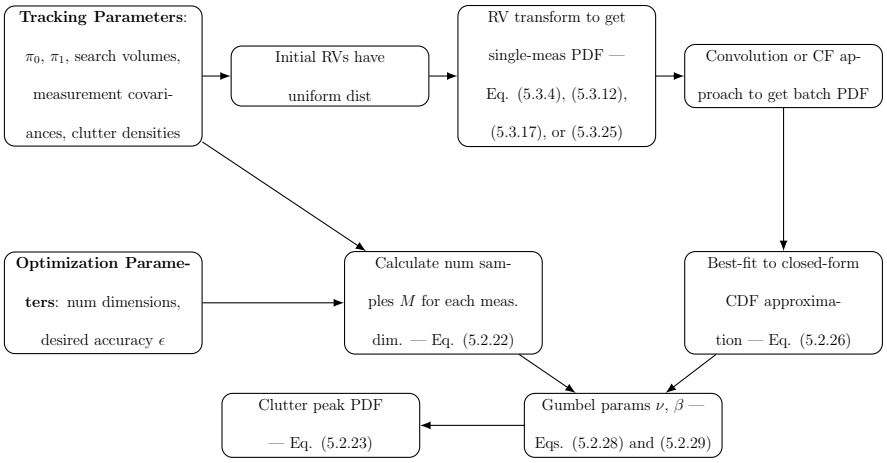

Fig. 5.3: Framework for determining clutter peak PDF

It is also instructive at this point to note the dependence of the Gumbel parameters and by extension, the ML-PMHT threshold, on the value of M_{tot}. The parameter ν, calculated in (5.2.28) is the location parameter in the Gumbel PDF — there is basically a one-to-one relationship between this and the ML-PMHT threshold. Typical values determined for l in (5.2.27), (5.2.28), and (5.2.29) are around $l \approx 2$. From this, the location parameter and the ML-PMHT threshold both increase approximately as $\sqrt{\ln M_{tot}}$.

For clarity, the overall framework for determining clutter peak EV PDFs is presented in Figure 5.3. This figure contains the relevant equation numbers to use for each step and is intended to be an overview of the ML-PMHT threshold value determination.

5.3 Extreme-value clutter PDFs

In this section, extreme-value (Gumbel) PDFs are calculated for four cases: one-dimensional measurements (bearing-only or time-delay-only), two-dimensional measurements (bearing and time-delay), three dimensional measurements (bearing, time-delay, and Doppler), or two-dimensional measurements with amplitude.

5.3.1 One-dimensional measurements

We start by developing the extreme-value peak clutter ML-PMHT LLR PDF for one-dimensional measurements, which in the multistatic active case, can be bearing-only or time-delay only. (The following result is not dependent on the type of measurement; it is just dependent on its dimensionality.) Start with the single-measurement one-dimensional LLR

$$\Lambda_1(z) = \ln\left\{1 + \frac{\pi_1}{\pi_0}\frac{V}{\sqrt{2\pi\sigma_1^2}}e^{-\frac{1}{2\sigma_1^2}(z-\mu)^2}\right\} \tag{5.3.1}$$

Now, simply treat the LLR as the transformation of a random variable

$$w \triangleq \Lambda_1(z) \tag{5.3.2}$$

where z is the original random variable, and w is the transformed random variable. Now μ is an arbitrary (fixed) target location somewhere in measurement space. If the PDF of z is given by $p_z(z)$, the PDF of w is given by [48]

$$p_w(w) = p_z[\Lambda_1^{-1}(w)]\left|\frac{dz}{dw}\right| \tag{5.3.3}$$

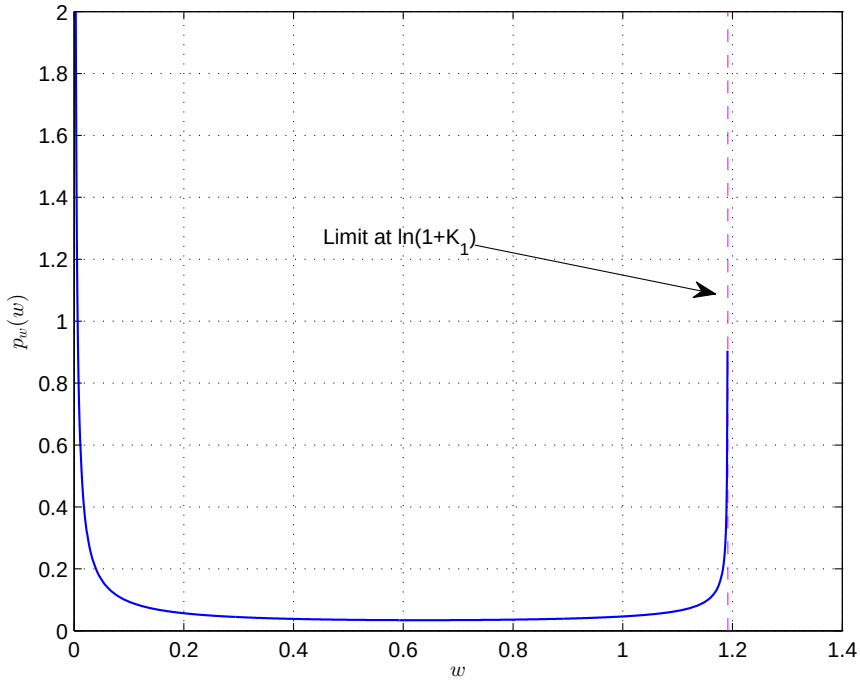

Fig. 5.4: 1-dimensional PDF $p_w(w)$ — single clutter measurement transformed by LLR function

Application of (5.3.3) to (5.3.1) produces the transformed PDF

$$p_w(w) = \frac{2}{V} \frac{\sigma_1 \exp(w)}{\sqrt{2 \ln \frac{1-\exp(w)}{K_1}}} \frac{1}{\exp(w) - 1} \qquad 0 \leq w \leq \ln(1 + K_1) \qquad (5.3.4)$$

where K_1 is given in (5.2.6). Figure 5.4 shows a plot of $p_w(w)$. Some interesting effects are present in this plot. The majority of the mass of the PDF is located at $w = 0$, meaning that on average, a clutter measurement will fall far enough away from the mean μ that the Gaussian term in (5.3.1) will be very small, so that overall the function is zero. However, near $w = \ln(1+K_1)$, there is a small "spike"

of probability. This is representative of a small but non-zero probability that some clutter measurements will fall near enough the mean μ that the Gaussian term in (5.3.1) makes a non-zero contribution to the LLR. One important thing to note is that the PDF described in (5.3.1) and shown in Figure 5.4 is independent of the target location μ. This is what allows the ML-PMHT LLR to be treated as a transformation of a uniformly distributed random variable.

At this point, we have the PDF for a single transformed clutter measurement – now we want the PDF for a batch of measurements, as represented by (5.2.1). This can done simply by adding the N IID single-measurement RVs (where N is the total number of measurements in the batch), each of which has the common PDF given by (5.3.4). Such a PDF can be calculated in two ways — either by directly convolving (5.3.4) with itself $N-1$ times, or calculating the characteristic function of (5.3.4), raising it to the power N, and then transforming this product back to a PDF. Both approaches were implemented and, as expected, produced identical results — direct convolution was simpler to implement but could become somewhat time-consuming, while the characteristic function approach (done using FFTs and inverse FFTs) was somewhat more involved to implement but faster.

These methods were applied to the PDF shown in Figure 5.4 for a value of $N = 600$. The resultant batch measurement PDF is shown in Figure 5.5. Although one would be tempted to apply the Central Limit Theorem and assume this resultant

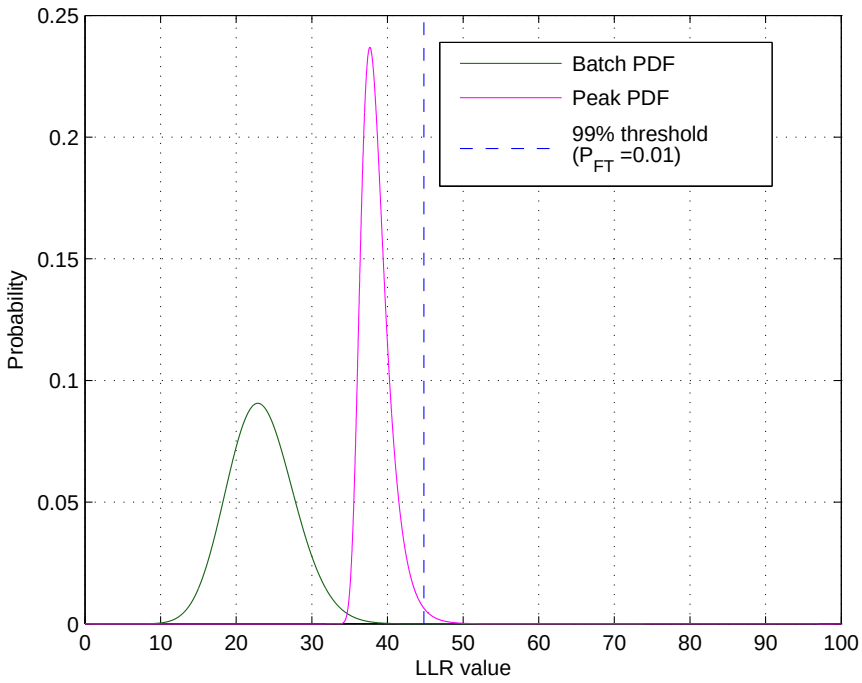

Fig. 5.5: 1-dimensional batch and extreme-value PDFs

PDF is Gaussian, it is incorrect to do so. The reason for this is that we will be concerned with the extreme values of this PDF, which have to do with the distribution's tail regions. The "true" PDF that is calculated by convolution does not match the PDF using CLT assumptions well enough in the tail region. Interestingly, the PDF shown in Figure 5.5 fits extremely well to a Nakagami distribution for most input parameters. However, this is not always the case — the relationship falls apart when the bearing measurement noise is small (less than 1 degree), so no use will be made of this observation. We will just use the PDF of the batch of measurements produced by either convolution or the characteristic function approach. (If the batch PDF were in general a Nakagami distribution it would be possible to go directly from the input tracking and clutter parameters to a tracking threshold.)

5.3.2 Two-dimensional measurements

Obtaining the PDFs for the case of two-dimensional measurements (bearing and delay-time) is more difficult than the one-dimensional case and must be done in steps. The two-dimensional random variable transformation for one set of measurements (again, the ML-PMHT LLR is being treated as simply a transformation) is given by

$$w_2 = \ln\left\{1 + \frac{\pi_1}{\pi_0}\frac{V}{\sqrt{|2\pi\mathbf{R}|}}\exp\left[-\frac{(z_1-\mu_1)^2}{2\sigma_1^2} - \frac{(z_2-\mu_2)^2}{2\sigma_2^2}\right]\right\} \tag{5.3.5}$$

It is first necessary in this case to obtain the PDF of the random variables in the exponent of (5.3.5) by making the following substitution

$$y_i = \left(\frac{z_i - \mu_i}{\sigma_i} \right)^2 \qquad i \in \{1, 2\} \tag{5.3.6}$$

Again, applying standard random variable transformation techniques gives the PDF of y_i

$$p_{y_i}(y_i) = \frac{\sigma_i}{V_i} y_i^{-\frac{1}{2}} \qquad 0 < y_i < \alpha_i \tag{5.3.7}$$

where α_i is given by

$$\alpha_i = \begin{cases} \frac{\mu_i^2}{\sigma_i^2} & \mu_i \geq \frac{V_i}{2} \\ \left(\frac{V_i - \mu_i}{\sigma_i} \right)^2 & \mu_i < \frac{V_i}{2} \end{cases} \tag{5.3.8}$$

Next, the PDF of $u_2 = y_1 + y_2$ can be determined through convolution; the result[5] of this is

$$p_{u_2}(u_2) = \begin{cases} \pi \frac{\sigma_1 \sigma_2}{V_1 V_2} & 0 \leq u_2 \leq \alpha_2 \\ \frac{\sigma_1 \sigma_2}{V_1 V_2} \left[\frac{\pi}{2} - \arcsin(1 - \frac{2\alpha_2}{u_2}) \right] & \alpha_2 < u_2 \leq \alpha_1 \\ \frac{\sigma_1 \sigma_2}{V_1 V_2} \left[\arcsin(\frac{2\alpha_1}{u_2} - 1) - \arcsin(1 - \frac{2\alpha_2}{u_2}) \right] & \alpha_1 < u_2 \leq \alpha_1 + \alpha_2 \end{cases} \tag{5.3.9}$$

[5] For the calculation of $p_{u_2}(u_2)$ we have set $\mu_i = V_i/2$. If we did not do this, a slightly more complex PDF would result — the first section of the PDF would have a uniform distribution, followed by *four* different arcsine segments. However, this assumption will not change the discussion in the sequel in any way — all of the arcsine sections will end up getting mapped to $w = 0$.

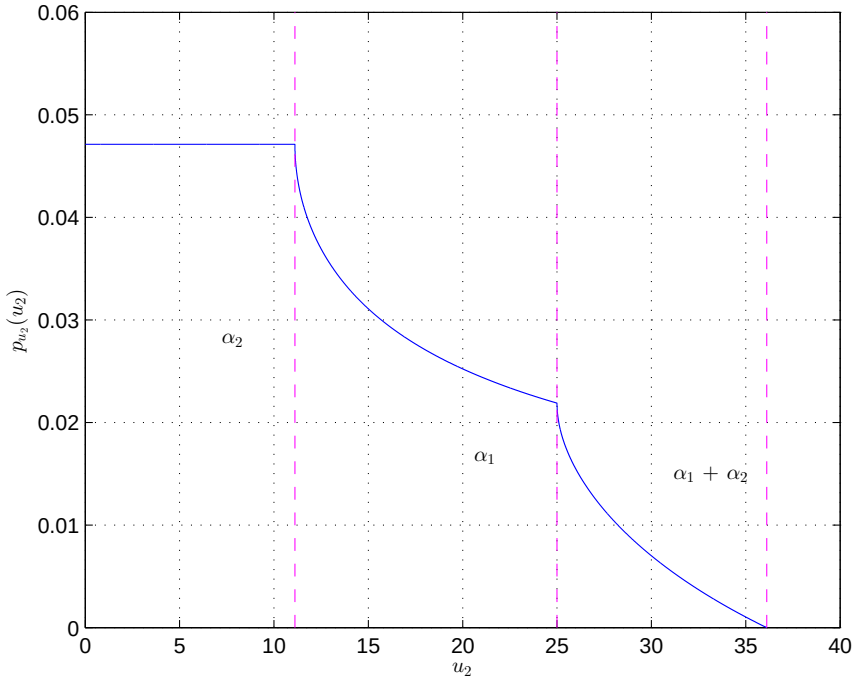

Fig. 5.6: 2-dimensional intermediate PDF of clutter measurement transformed by LLR function

This calculation requires that $\alpha_2 < \alpha_1$, but since convolution is commutative, this condition can be assumed to hold without loss of generality. The PDF described by (5.3.9) is plotted in Figure 5.6. (Note that this PDF is only an "intermediate" step — it is not the final transformation of a measurement random variable to a random variable that represents the ML-PMHT LLR.) At first glance, this result contradicts the previous assertion that the (arbitrary) means fall out of the calculations; they are present in both the limit and the PDF in (5.3.9). However, as will be seen shortly, they do drop out of the expressions after the next calculation.

At this point, one final transformation is applied:

$$w_2 = \ln\left[1 + K_2 \exp(-u_2/2)\right] \tag{5.3.10}$$

where K_2 is given by

$$K_2 = \frac{\pi_1}{\pi_0} \frac{V_1 V_2}{\sqrt{|2\pi\mathbf{R}|}} \tag{5.3.11}$$

What is important about this transformation is how it maps the positive u_2 axis to the positive w_2 axis. For all values of u_2 greater than some reasonably small u_{lim}, the transformation in (5.3.10) maps u_2 to $w_2 = 0$. Thus, all but the very left side of the PDF shown in Figure 5.6 is transformed to a delta function at the origin in w_2-space. This effect is shown in Figure 5.7. For realistic ML-PMHT parameter values, u_{lim} is much less than all the α_i values, so again, the μ_i values disappear from the final result.

As a result of this mapping, a two-step process is required to obtain the PDF for w_2, which represents the ML-PMHT LLR for a single clutter measurement in the 2-D case. First, the portion of the PDF of u_2 going from $u_2 = 0$ to $u_2 = \alpha_2$ is transformed to a (partial) PDF for w_2. This is integrated to find the mass of probability that is not at $w_2 = 0$. The remainder of the probability mass is then assigned as a delta function at $w_2 = 0$ so the PDF of w_2 integrates to 1. The resultant PDF is

$$p_{w_2}(w_2) = \begin{cases} C\delta(0) & w_2 = 0 \\ 2\pi \frac{\sigma_1 \sigma_2}{V_1 V_2} \frac{\exp(w_2)}{\exp(w_2)-1} & 0 < w_2 \leq \ln(1 + K_2) \end{cases} \tag{5.3.12}$$

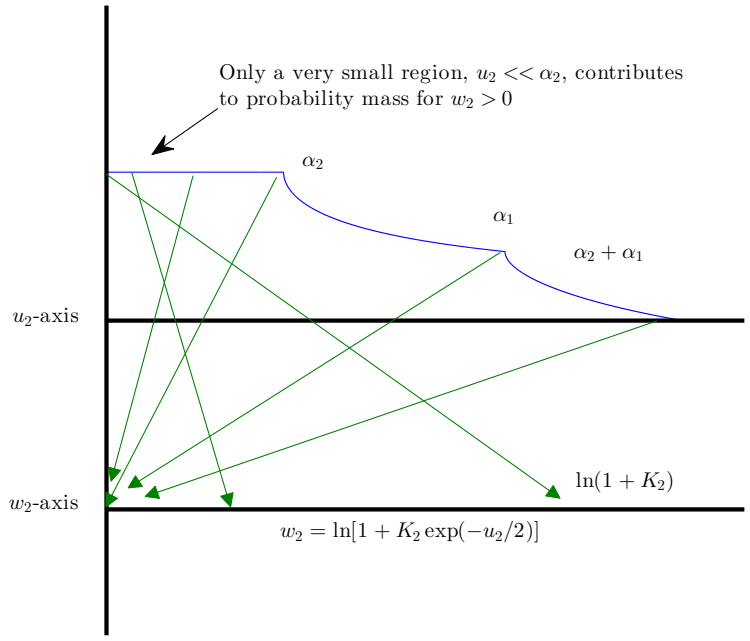

Fig. 5.7: Transformation of random variables from u_2 to w_2

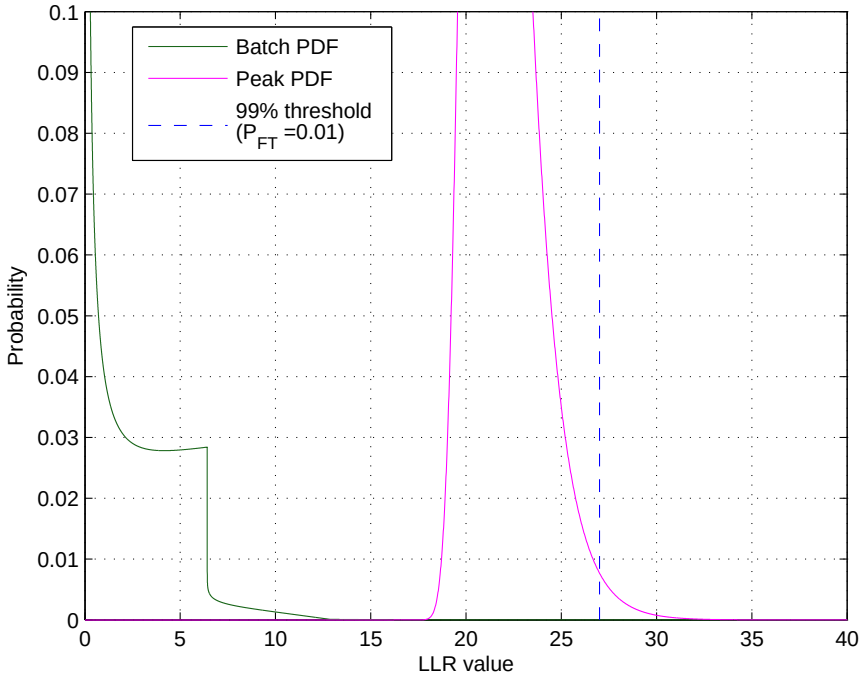

Fig. 5.8: Two-dimensional ML-PMHT PDFs

where δ is a Dirac delta function and C is a normalization constant that is scaled so that the entire PDF integrates to 1. This PDF represents the ML-PMHT LLR for a single 2-D measurement. Now, with either convolutions or the characteristic function approach (again, they produce the same results), we can obtain the PDF that represents the ML-PMHT LLR for a batch of data (in this and all following cases $N = 108$ as opposed to the 1-D case of $N = 600$). This is shown in Figure 5.8. A comment on the shape of this batch PDF is in order. First, only a small fraction of the total probability mass is located at a value of w_2 that is non-zero – in the case of Figure 5.8 the non-zero mass only accounts for approximately 20 percent

of the total probability. This makes sense – for a clutter measurement to have a non-zero effect on the ML-PMHT LLR, the measurements in *both* dimensions have to fall close to mean of the arbitrary Gaussian. In the 1-D case, there was (obviously) only one dimension in which the clutter had to be close to the arbitrary mean.

Finally, the same technique that was used in the 1-dimensional case to find the PDF describing the largest peak point in the LLR was used. First, the equivalent number of samples M_{tot} was determined; again this is the number of times the LLR surface should be sampled to obtain the same accuracy as the actual numerical optimization. Note that in this case, M_{tot} will be the product of the required number of samples in each of the two (independent) measurement dimensions. Next, the CDF of w_2 is obtained by numerically integrating the batch PDF. This CDF is then approximated with (5.2.26), and finally, the Gumbel distribution parameters are obtained via (5.2.28) and (5.2.29). This PDF is shown in Figure 5.8.

5.3.3 Three-dimensional measurements

The process of obtaining the PDFs for 3-dimensional measurements (e.g. bearing, time-delay, and Doppler, or range, azimuth, and elevation) is very similar to the process for obtaining the PDFs for 2-dimensional measurements. The same steps are followed — first obtaining PDFs of the shifted, scaled and squared uniform

random variables, which in this case are denoted y_1, y_2 and y_3. Then, these three RVs added together such that

$$u_3 = y_1 + y_2 + y_3 \tag{5.3.13}$$

The PDF of u_3 is obtained by convolving (5.3.7) with itself two times; this convolution does not have a closed form solution except for the area from $0 \leq u_3 \leq \alpha_3$, where $\alpha_3 < \alpha_2 < \alpha_1$. The (partial) PDF in this region is given by

$$p_{u_3}(u_3) = 2\pi \frac{\sigma_1 \sigma_2 \sigma_3}{V_1 V_2 V_3} \sqrt{u_3} \qquad 0 \leq u_3 \leq \alpha_3 \tag{5.3.14}$$

Again, for $u_3 > \alpha_3$, the PDF $p_{u_3}(u_3)$ is non-zero, but it cannot be expressed in closed form due to having to convolve a square root with an arcsine function. However, as was the case with the 2-D measurement, this does not matter. When transforming from u_3 to w_3, only a small percentage of the support of u_3, $0 \leq u_3 \leq u_{max}$, where $u_{max} \ll \{\alpha_1, \alpha_2, \alpha_3\}$ gets mapped to non-zero values of w_3. This transformation is essentially the same as in (5.3.10), namely,

$$w_3 = \ln\left[1 + K_3 \exp(-u_3/2)\right] \tag{5.3.15}$$

with

$$K_3 = \frac{\pi_1}{\pi_0} \frac{V_1 V_2 V_3}{\sqrt{|2\pi \mathbf{R}|}} \tag{5.3.16}$$

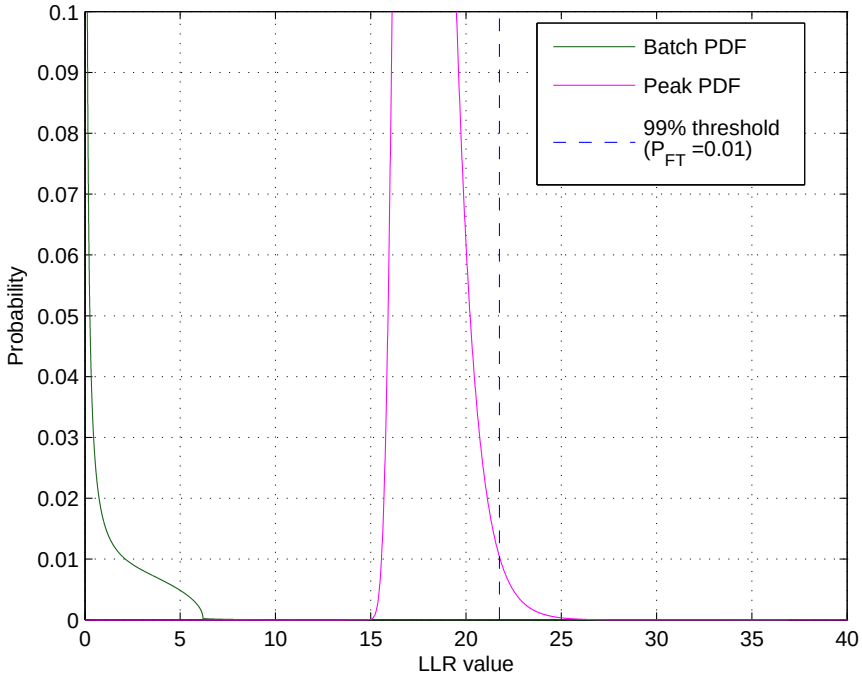

Fig. 5.9: Three-dimensional ML-PMHT PDFs

where now $\mathbf{R}$ is a 3x3 measurement covariance matrix. The resultant PDF that represents the single-measurement LLR is

$$p_{w_3}(w_3) = \begin{cases} C\delta(0) & w_3 = 0 \\ 4\pi\sqrt{2}\frac{\sigma_1\sigma_2\sigma_3}{V_1V_2V_3}\frac{\exp(w_3)}{\exp(w_3)-1}\sqrt{\ln\left(\frac{K_3}{\exp(w_3)-1}\right)} & 0 < w_3 \leq \ln(1+K_3) \end{cases} \tag{5.3.17}$$

where again, C is a constant such that the PDF integrates to one.

Again, this single-measurement PDF is used to calculate the batch ML-PMHT PDF via either repeated convolutions or the characteristic function approach with identical results. The resultant batch PDF is shown in Figure 5.9.

This figure has the same scale and axes as the 2-D case — note that the third dimension (e.g. Doppler processing) further reduces the amount of mass that is present in the PDF for $w_3 \neq 0$. In the 2-D case, approximately 20 percent of the probability mass was located away from $w_2 = 0$; when adding Doppler, less then 10 percent of the probability mass is located at non-zero values of w_3. This translates directly into clutter measurements causing lower peaks on the overall ML-PMHT LLR.

Again, now that the PDF representing the entire batch ML-PMHT LLR is available, the process is the same to generate the extreme value PDF that describes the peak value of the LLR caused by clutter. First, the representative number of samples M_{tot} is determined; now M_{tot} is a product of the required number of samples in each of the three measurement dimensions. Next, the CDF representing the overall batch ML-PMHT LLR is computed by numerically integrating the PDF, and this CDF is then approximated with the closed-form expression from (5.2.26). Lastly, the Gumbel distribution parameters are obtained via (5.2.28) and (5.2.29). This Gumbel distribution is shown in Figure 5.9. Note that the addition of Doppler information has, as expected, shifted the peak clutter PDF to the left.

5.3.4 Two-dimensional measurements plus amplitude

We develop the peak clutter PDF for one last ML-PMHT combination — two dimensional measurements (time-delay and azimuth) plus the amplitude feature. The ML-PMHT LLR in this case must include $\rho_j(i)$, the amplitude likelihood ratio, as is shown in (5.2.1). The clutter amplitude is assumed to have a Rayleigh distribution [41]; in intensity space, this becomes an exponential distribution. The feature likelihood ratio, in intensity space, is given by

$$\rho_j(i) = \frac{1}{\sigma^2} \frac{e^{\frac{\tau - v}{\sigma^2}}}{e^{\tau - v}} \qquad v \geq \tau \tag{5.3.18}$$

where v is the intensity, σ^2 is the expected intensity, and τ is the detector threshold (again, in units of intensity). Now, (5.3.18) can be incorporated into (5.3.5) in the following manner by combining all random variables inside the exponential

$$w_{2a} = \Lambda_{2a}(z_1, z_2, v) = \ln(1 + K_2' e^{-\frac{1}{2} u_{2a}}) \tag{5.3.19}$$

where

$$K_2' = \frac{K_2}{\sigma^2} e^{K_\sigma \tau} \tag{5.3.20}$$

and

$$K_\sigma = \frac{1 - \sigma^2}{\sigma^2} \tag{5.3.21}$$

Finally, the random variable u_{2a} in the exponent is given by

$$u_{2a} = 2K_\sigma v + y_1 + y_2 \tag{5.3.22}$$

where the y_i RVs are defined in (5.3.6). The PDF for v is just an exponential; the PDF for $v' = 2K_\sigma v$ is given by

$$p_{v'}(v') = \frac{e^\tau}{2|K_\sigma|} e^{\frac{v'}{2|K_\sigma|}} \qquad v' < -2|K_\sigma|\tau \tag{5.3.23}$$

The partial PDF for u_{2a} (where $u_{2a} < \alpha_2$) can be obtained by convolving the PDF shown in Figure 5.6 with the PDF in (5.3.23). Again, there is no need to calculate the PDF for $y_{2a} > \alpha_2$ since these values will get mapped to $w_{2a} = 0$. This partial PDF is

$$p_{u_{2a}}(u_{2a}) = \begin{cases} \frac{\pi\sigma_1\sigma_2}{V_1V_2} e^\tau e^{\frac{u_{2a}}{2|K_\sigma|}} & -\infty < u_{2a} < -2|K_\sigma|\tau \\ \\ \frac{\pi\sigma_1\sigma_2}{V_1V_2} & -2|K_\sigma|\tau < u_{2a} \lesssim \alpha_2 \end{cases} \tag{5.3.24}$$

There are two things of note here — first the PDF of u_{2a} contains some small but non-zero probability mass for $u_{2a} < 0$ (the PDF for u_2 with no amplitude did not). This will end up affecting the PDF of the right-tail of the next random variable transformation. Next, the PDF of u_{2a} does not go as a "boxcar" function to α_2 — it starts to exponentially decay slightly before that. However, this occurs in the area of support for the PDF that maps to $w_{2a} = 0$, so we can ignore the exact structure of the PDF in this region.

Again, this PDF is put through an RV transformation similar to (5.3.10). The

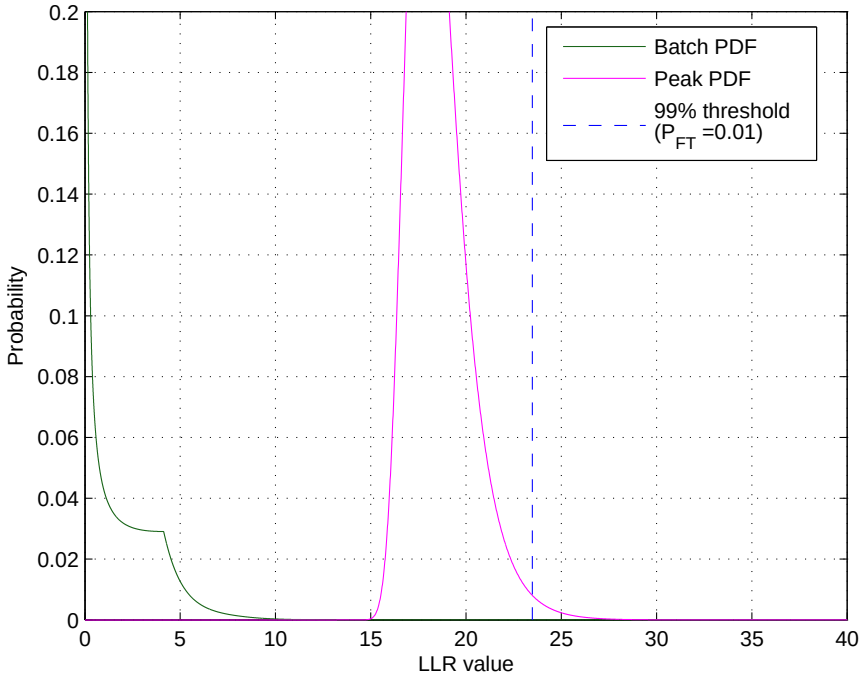

Fig. 5.10: Two-dimensional ML-PMHT PDFs with amplitude

resultant PDF is

$$p_{w_{2a}}(w_{2a}) =$$

$$\begin{cases} C\delta(0) & w_{2a} = 0 \\ 2\pi\frac{\sigma_1\sigma_2}{V_1V_2}\left[1 - e^{\tau}\left(\frac{\exp(w_{2a})-1}{K'}\right)^{\frac{1}{K_\sigma}}\right]\frac{\exp(w_{2a})}{\exp(w_{2a})-1} & 0 < w_{2a} \leq \ln\left(1 + K_2' e^{-K_\sigma \tau}\right) \end{cases} \tag{5.3.25}$$

Again, the batch PDF that results from convolving this expression is shown in Figure 5.10. It is instructive to compare this with Figure 5.8 — when using the amplitude feature, the mass of the PDF is clearly shifted to the left (i.e. towards zero), meaning that the overall ML-PMHT LLR is less affected by clutter when

using the amplitude feature. However, there is an important caveat here. The value of σ^2 — the expected intensity of the amplitude from the target — is treated as a constant in this expression. This is implicitly assuming that this expected value is both constant and known. First, this expected value will often vary from measurement to measurement, especially if multiple source-receiver pairs are being used in the problem. Secondly, this value is a function of many things, but especially of acoustic transmission loss from source to target to receiver. This transmission loss can be predicted or modeled, but the underwater ocean environment can make doing this accurately very difficult. A more accurate treatment of this should probably treat the expected intensity as a random variable; however, this would preclude the closed-form analysis that led to the development of (5.3.25).

5.4 Threshold determination

At this point, the extreme value distributions (peak PDFs) from clutter are used to generate tracking thresholds. These results are compared to thresholds developed via the (much slower) method described in [18] — i.e. repeatedly simulating and optimizing a (clutter-only) ML-PMHT LLR surface.

For a Gumbel distribution, the CDF (and a threshold) is easily calculable once the parameters for the PDF are known. For a Gumbel PDF given by (5.2.23), the

CDF is given by

$$F(x) = \exp\left(-e^{-\frac{x-\nu}{\beta}}\right) \tag{5.4.1}$$

Then, for a desired probability of false track acceptance L, the threshold κ is found by inverting the CDF

$$\kappa = \nu - \beta \ln\left(\ln \frac{1}{1-L}\right) \tag{5.4.2}$$

The problem parameters that were used to calculate "model" thresholds are listed in Table 5.1 (they are also plotted for their respective scenarios in all the above clutter PDF plots). Next, results from the model threshold determination and actual Monte Carlo testing are shown in Table 5.2. The model-based threshold values are very close to the actual empirical threshold values obtained from Monte Carlo testing. It was far faster to calculate the model-based values — each only needed only on the order of several seconds. In contrast, the actual values obtained by simulating and optimizing multiple clutter surfaces were much more time-consuming to calculate — for 5000 Monte Carlo runs, it took on the order of 10 hours to obtain each value.

It should be further stressed that this approach is *not* unique to ML-PMHT. There is no reason why this technique should not work for any maximum likelihood approach. As long as the underlying distributions of the measurements are known or assumed, any likelihood ratio can be treated as just a random variable transformation, and the methods above can be used. While in many cases, the

Table 5.1: Values used for threshold determination

1-Dimensional Parameters	
Angular volume	180°
Angular error	2°
Ave meas per scan	10
Number of scans	60
π_1	0.05
π_0	0.95
2-D and 3-D Parameters	
Angular volume	360°
Angular error	5°
Time delay volume	60 sec
Time delay error	0.1 sec
Range-rate volume	30 units per sec
Range-rate error	0.5 units per sec
Amplitude threshold	7 dB
Expected target amplitude	10 dB
Ave meas per scan	9.8
Number of scans	11
π_1	0.15
π_0	0.85

Table 5.2: Comparison of model vs. empirical tracking thresholds (dimensionless units for LLR)

	Model	Empirical	Empirical 95% Confidence Interval
1-D	41.8	42.0	[41.9, 42.2]
2-D	27.0	27.3	[26.7, 27.9]
3-D	21.8	21.9	[21.5, 22.4]
2-D amp	23.5	23.0	[22.5, 23.6]

calculations will likely have to be done numerically rather than in closed form, there are no assumptions in the above approach that would preclude its more general application to any maximum likelihood technique.

5.5 Conclusions

We have presented a novel method for determining the PDFs describing the maximum points in the ML-PMHT LLR caused by clutter using random variable transformations. The EV distribution describing the peak point (global maximum) in the LLR due to clutter can be used to quickly and easily determine a tracking threshold; previously, this had to be done either through trial-and-error or with very time consuming optimizations of Monte Carlo simulations. Thresh-

olds obtained in this manner match up extremely well with thresholds obtained by (the much more time consuming) Monte-Carlo simulation and optimization of ML-PMHT LLR surfaces. The ability to rapidly and accurately determine tracking thresholds add significant capability to the ML-PMHT multistatic active tracking framework.

Chapter 6

Probability of Track Detection and Target Trackability

6.1 Introduction

This chapter is very closely related to Chapter 5. In that chapter, we developed a framework for calculating the peak probability density function (PDF) of the ML-PMHT LLR that was due to clutter. This peak clutter PDF was then used to obtain a tracking threshold in a rapid and accurate fashion. The idea underlying the framework was to treat the ML-PMHT LLR as a function, and the clutter measurements as uniformly-distributed random variables. The ML-PMHT LLR then just became a random-variable transformation. First the PDF representing a single measurement was developed, and then via convolution or a characteristic function approach, a "batch" ML-PMHT PDF was obtained. This batch PDF represented the LLR at *any* point in space. In order to determine the "peak" PDF (the maximum point on the LLR due to clutter), extreme-value theory [22], [24], [28], [30], [38] was used. With the peak PDF due to clutter determined, the Neyman-Pearson lemma [49] was used to obtain a clutter threshold that produced

a certain false-alarm rate.

Now, we employ a similar technique to develop an expression for the maximum value of the ML-PMHT LLR that is caused by a target. The process is slightly different because while clutter measurements are assumed to be uniformly distributed in the search volume, target measurements are assumed to be normally distributed around the actual target location. Extreme value theory is not required in the target case to obtain the peak PDF; by working with measurements distributed around the true target location, we are already creating the PDF that describes the peak in the LLR created by the target. Now, with expressions for the peak clutter PDFs (developed above in Chapter 5 and in [61]) and peak target PDFs, we can develop "Tracker Operating Characteristic" (TOC) curves, similar to Receiver Operating Characteristic (ROC) curves found in detection theory, which will provide insight into whether or not a given target can be tracked in a certain environment.

In a related effort, a central limit theorem approach was used in [40] and [41] to obtain the probability of target track detection by approximating the target LLR PDF with a moment-matched Gaussian. However, as will be shown below, the exact target LLR PDF is not symmetric; as a result, this work should provide more accurate results.

6.2 Overall framework for LLR extreme value PDF determination

Here, we present the framework used for going from the parameters that are present in the ML-PMHT LLR to determining the distributions of the maximum (peak) points in the LLR caused by target measurements.

6.2.1 Framework flow

We summarize briefly the steps necessary to obtain peak PDFs caused by a target. These steps are simpler than those necessary to calculate the peak PDFs due to clutter. First, the PDF formed by a single target measurement is calculated. Then the single-measurement PDF and the ML-PMHT assumption of measurement independence are used to derive the batch PDF. As opposed to the clutter case, the batch PDF *is* the peak PDF, since we are simulating only around one point, the true target location, which is already the peak point. There is no need to employ extreme-value theory in the case of target measurements.

Peak probability density functions are developed for four specific cases: one-dimensional measurements (either bearings-only or time-delay only), two-dimensional measurements (bearing and time-delay), three-dimensional measurements (bearing, time-delay, and Doppler), and two-dimensional measurements with amplitude.

6.2.2 Calculating the batch PDF (overall target likelihood) via single-measurement transformation

As with clutter, the first step in the process is to calculate the PDF that represents a batch of measurements. The necessary background discussing the development of and assumptions behind the ML-PMHT LLR was covered in Chapter 5 and is omitted here. We go directly to the ML-PMHT LLR for a single (one-dimensional) measurement, which is written as

$$\Lambda_1(z) = \ln\left\{1 + \frac{\pi_1}{\pi_0}\frac{V}{\sqrt{2\pi\sigma_1^2}}e^{-\frac{1}{2\sigma_1^2}[z-\mu(k)]^2}\right\} \tag{6.2.1}$$

where π_1 is the prior probability that a given measurement is from the target, π_0 is the prior probability that a given measurement is from clutter, V is the search volume, σ_1 is the measurement noise standard deviation, and $\mu(k)$ is the projected target location in measurement space at time update k (this is a function of $\mathbf{x}$, the target parameter vector).

As in the case with clutter, the key point is to treat this single-measurement LLR as just a function that transforms a random variable (RV) — if the measurement z in (6.2.1) is treated as a random variable, normally distributed around some assumed target location,[1] then $t = \Lambda_1(z)$, where t is just a new transformed variable. If the PDF of z is given by $p_z(z)$, the PDF of t is given (for a scalar z)

[1] If there is a clutter measurement closer to $\mu(k)$, this increases the likelihood; ignoring this, we have a slightly conservative approach.

by [48]

$$p_t(t) = p_z[\Lambda_1^{-1}(t)] \left| \frac{dz}{dt} \right| \tag{6.2.2}$$

Again, it is possible to calculate in closed-form the single-measurement PDF for the four cases mentioned – scalar measurements, 2-D measurements, 3-D measurements, and 2-D measurements with amplitude. With these single-measurement PDFs available, the batch PDF can be calculated. From the ML-PMHT assumptions, each measurement is independent. Additionally, as will be shown in Section 6.3, the individual measurement PDFs are only a function of the fixed ML-PMHT parameters (π_0, π_1, V, σ_1) — they are *not* functions of the target parameter vector $\mathbf{x}$ or the resultant projected target location $\mu(k)$ in measurement space, and thus are IID. (If each single measurement PDF was a function of $\mu(k)$, which generally changes from scan-to-scan, the individual transformed measurements would not have identical distributions.) Due to the fact that they are IID and the overall log-likelihood is the sum of single-measurement log-likelihoods, i.e. $\Lambda(z) = \sum_{i=1} \Lambda_i(z)$, it is possible to use convolution to obtain the PDF for a batch of measurements. If there are N_t target measurements in a batch, then the single-measurement PDF can be convolved with itself $N_t - 1$ times to get the batch PDF. Alternatively, we could calculate the characteristic function (CF) of the single measurement PDF, raise it to the power of N_t, and then take the inverse characteristic function — this will produce the same results.

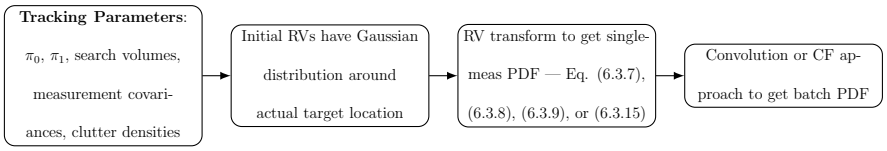

Fig. 6.1: Framework for determining target likelihood PDF

Finally, we are simulating measurements around the true target location, which by its construction, is already the peak point on the LLR caused by the target. There is no need to apply extreme value theory, as there was in the case of clutter measurements.

The (relatively simple) framework for determining the peak PDF due to target measurements is shown in Figure 6.1.

6.3 Target batch PDFs

At this point, we actually develop the peak PDFs for the target case. Again, this is significantly simpler than it was for the case of clutter. For the target, Gaussian measurements are assumed to be distributed around the actual target location. The batch PDF (target likelihood) that results from this does not represent the batch PDF over the entire parameter space, but instead just the PDF conditioned on the target location. Thus, in the case of the target, the batch PDF *is* the peak PDF, so no further work is required. Secondly, since the target measurements are Gaussian distributed about the true target state, the random variable trans-

formations that are needed become simpler. The terms in the exponent of the transformations (see (6.2.1) as an example) can be written as

$$u_i = v_i^2 \tag{6.3.1}$$

where

$$v_i = \frac{x_i - \mu_i}{\sigma_i} \tag{6.3.2}$$

In the clutter case, x_i was uniformly distributed, so v_i was a shifted and scaled uniform RV. Squaring and then adding such RVs together produced a complicated transformed RV; in some cases, no closed form solution was available. However, in the target case, x_i is a Gaussian random variable[2] with mean μ_i and standard deviation σ_i. Thus, the transformation described by (6.3.2) is by inspection a $\mathcal{N}(0,1)$ random variable. Squaring and summing such random variables in the case of multidimensional measurements, as will be discussed below, is significantly easier than it was in the case of clutter.

6.3.1 General case for target measurements with no amplitude information

For the case of target measurements with either one, two, or three dimensions, we can discuss the general transformation for the peak target PDF. The random

[2] For clarity, this is written assuming the individual x_i RVs are independent. However, as will be shown below, for a general case, this assumption of independence is not necessary.

variable transformation is given by

$$t_d = \Lambda_d(\mathbf{z}) = \ln\left\{1 + K_d e^{-\frac{1}{2}w_d}\right\} \qquad d = \{1, 2, 3\} \tag{6.3.3}$$

where d denotes the number of dimensions, and K_d is given by

$$K_d = \frac{\pi_1}{\pi_0}\frac{V_d}{\sqrt{|2\pi\mathbf{R}|}} \tag{6.3.4}$$

Here V_d is the measurement volume, and $\mathbf{R}$ is the measurement covariance matrix. Finally, w_d is

$$w_d = \boldsymbol{x}^T\mathbf{R}^{-1}\boldsymbol{x} \tag{6.3.5}$$

where $\boldsymbol{x}$ is a vector containing the component (Gaussian) x_i RVs. By inspection then, w_d is a chi-squared RV with d degrees of freedom. (Note that here in the general case there is no requirement for independence of the component x_i RVs.) Putting such an RV through the transformation described by (6.2.2) gives the final output PDFs. For d degrees of freedom, the PDF of t_d is given by

$$p_{t_d}(t_d) = \frac{\exp(t)}{K_d\Gamma(d/2)}\left\{\ln\left[\frac{K_d}{\exp(t)-1}\right]\right\}^{d/2-1} \qquad 0 \le t_d \le \ln(1+K_d) \tag{6.3.6}$$

The resultant transformed single-measurement PDF in the one-dimensional case is

$$p_{t_1}(t_1) = \frac{1}{K_1\sqrt{\pi}}\frac{\exp(t_1)}{\sqrt{\ln\frac{K_1}{\exp(t_1)-1}}} \qquad 0 \le t_1 \le \ln(1+K_1) \tag{6.3.7}$$

This PDF is plotted in Figure 6.2. (The parameters used for this plot as well as the following plots in this section are given in Table 6.1.) Note that $p_{t_1}(t_1)$ is *not*

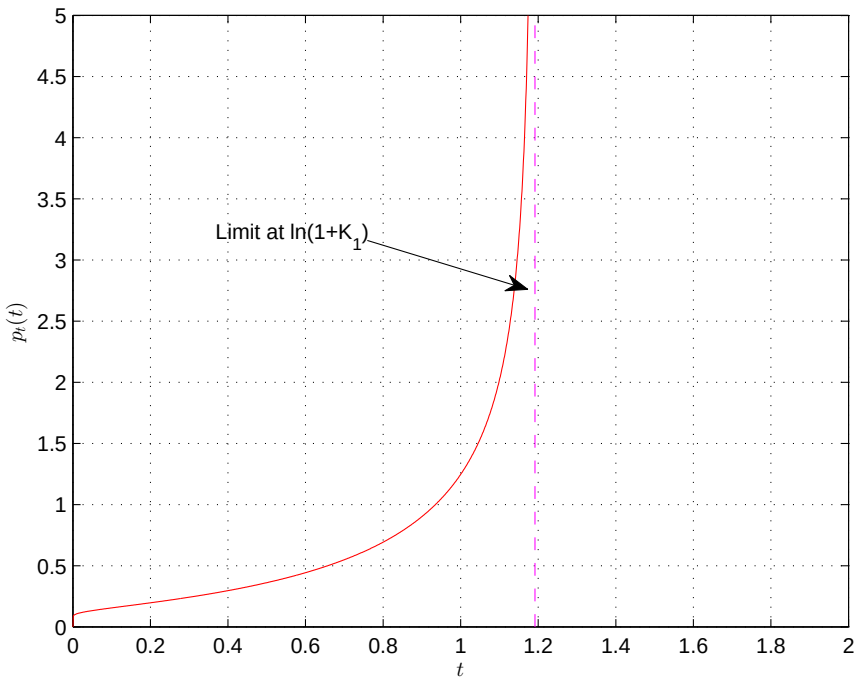

Fig. 6.2: One-dimensional single-measurement PDF (target likelihood)

a function of $\mu(k)$ – this is what allows the individual measurement PDFs to be treated as IID from scan to scan. As a result of this, (6.3.7) can be convolved with itself $N_t - 1$ times,[3] where N_t is the expected number of target measurements. (The method for calculating N_t will be discussed in Section 6.3.3.) The resultant batch PDF, which now is also the peak PDF for the target, is shown in Figure 6.3. In this as well as subsequent figures, the ML-PMHT threshold, calculated from the peak clutter PDF developed in Chapter 5, is also plotted.

In the two-dimensional case, a surprisingly simple expression results for the transformed single-measurement PDF:

$$p_{t_2}(t_2) = \frac{\exp(t_2)}{K_2} \qquad 0 \leq t_2 \leq \ln(1 + K_2) \tag{6.3.8}$$

One again, note that the single-measurement PDF is independent of the target parameter $\mu(k)$. This PDF is plotted in Figure 6.4. It is convolved with itself $N_t - 1$ times, and the resultant batch PDF is plotted in Figure 6.5. Again, this batch PDF is also the peak target PDF.

Finally, the transformed three-dimensional single-measurement PDF is given by

$$p_{t_3}(t_3) = \frac{\exp(t_3)}{\Gamma(\frac{3}{2})K_3}\sqrt{\ln\frac{K_3}{\exp(t_3) - 1}} \qquad 0 \leq t_3 \leq \ln(1 + K_3) \tag{6.3.9}$$

This PDF is plotted in Figure 6.6. Once again, it is independent of $\mu(k)$, so

[3] This can easily be done with Matlab's *conv* function. Alternatively, this can be obtained using exponentiation of the characteristic function. The latter is preferable in the case where many measurement LLRs need to be summed together.

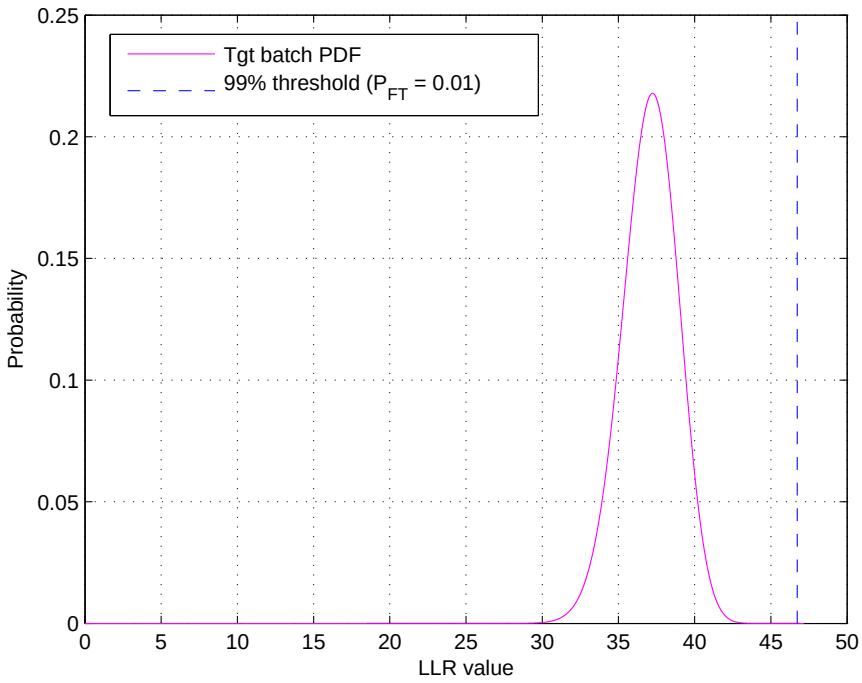

Fig. 6.3: One-dimensional ML-PMHT target batch PDF (overall target likelihood). The threshold calculated from the peak clutter PDF is also plotted.

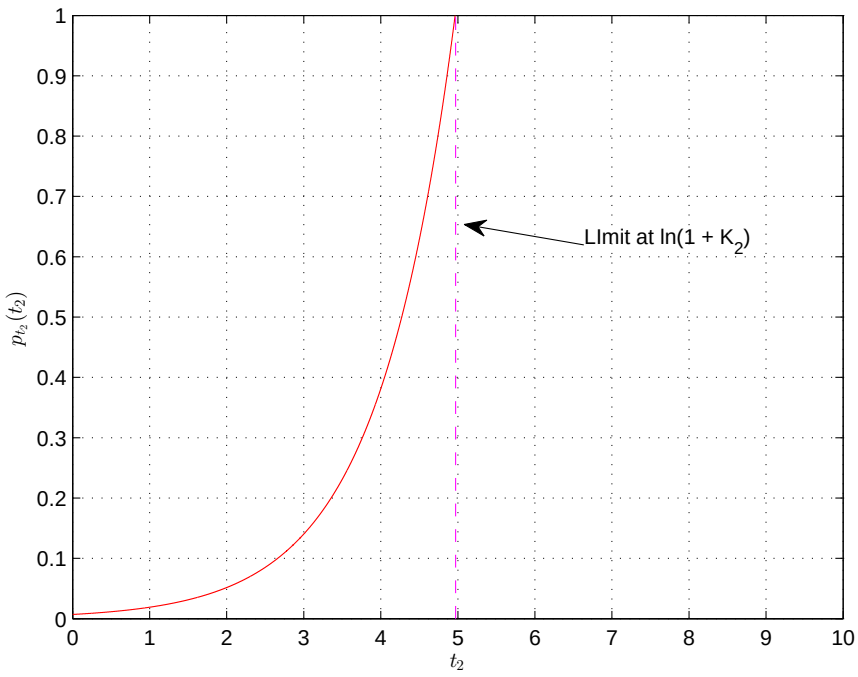

Fig. 6.4: Two-dimensional single-measurement PDF (target likelihood)

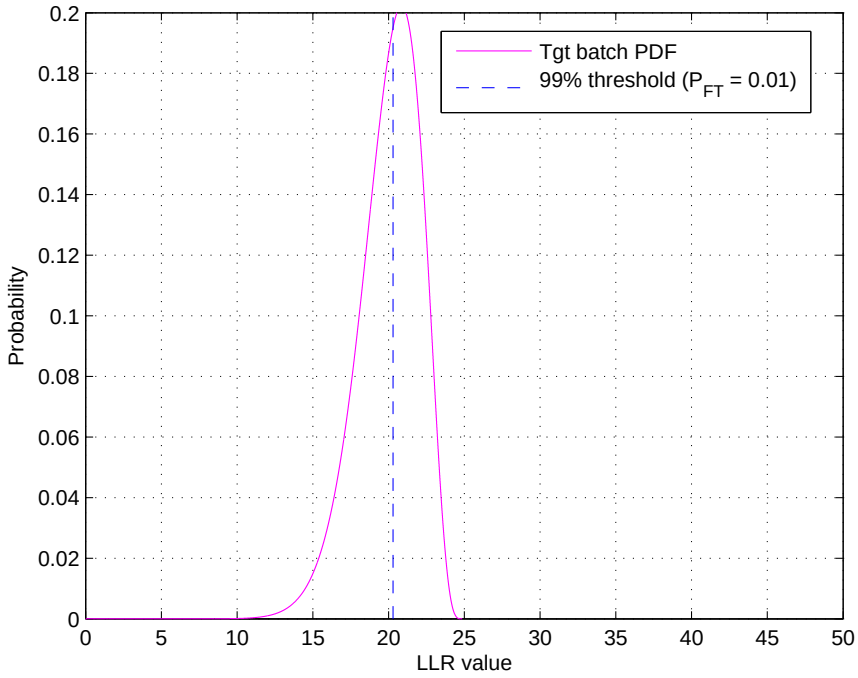

Fig. 6.5: Two-dimensional ML-PMHT target batch PDF (overall target likelihood). The threshold calculated from the peak clutter PDF is also plotted.

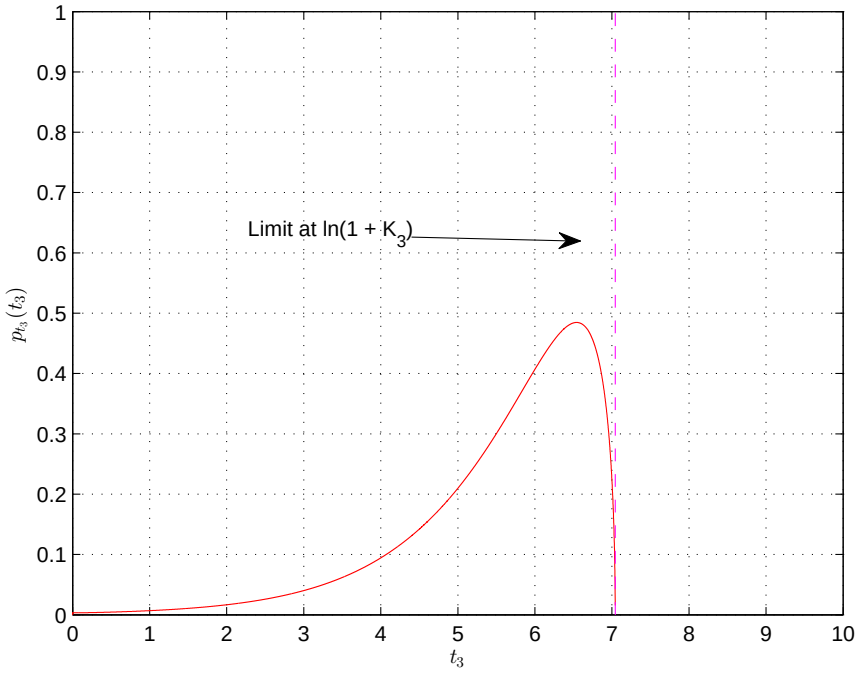

Fig. 6.6: Three-dimensional single-measurement PDF (target likelihood)

each transformed target RV can be treated as being independent. The PDF is convolved with itself $N_t - 1$ times, and the result is shown in Figure 6.7.

At this point, it instructive to compare the single-measurement and batch/peak plots for the 3-D case (Figures 6.6 and 6.7) with those of the 2-D case (Figures 6.4 and 6.5). All parameters in the 2-D and 3-D cases are the same, with the exception of the addition of the third dimension parameters — i.e. the only differences between the two should be due to the addition of Doppler information. The target batch PDF for the 3-D case is shifted to the right, as compared to the two-dimensional case. This is a first (expected) indication that adding Doppler

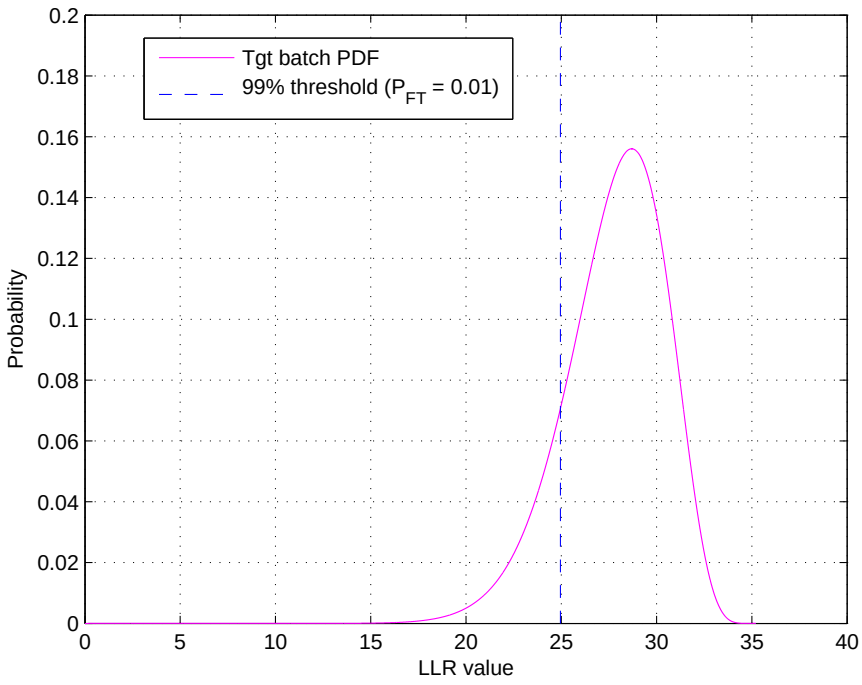

Fig. 6.7: Three-dimensional ML-PMHT target batch PDF (overall target likelihood). The threshold calculated from the peak clutter PDF is also plotted.

increases the peak in the ML-PMHT LLR due to the target, and thus would seem to increase "trackability." Some care must be taken here — the true measure of trackability will be between the peak of the target distribution and the clutter thresholds (calculated via the methods in Chapter 5). This distance does gets bigger in the 3-D case, which shows that, as expected, processing Doppler (in this example) improves trackability. Very interestingly, it turns out that this is not always the case — this will be discussed in much more detail in Section 6.4.

6.3.2 Two-dimensional target measurements with amplitude information

We again return to the two-dimensional case for the target, but now with amplitude information present in the LLR. This transformation does not fit with the general case described by (6.3.3), so it is described separately here. The target amplitude has a Rayleigh distribution following the work of [41]; for mathematical convenience, we elect to operate in intensity space, so the target returns have a thresholded exponential distribution described by

$$f_1^{\tau}(\varsigma) = \frac{1}{\sigma^2} e^{\frac{\tau-\varsigma}{\sigma^2}} \qquad \varsigma > \tau \tag{6.3.10}$$

The manipulations to get the PDF representing the single-measurement ML-PMHT LLR are very similar to those developed in Chapter 5; we summarize

them here. First, define K_2' as

$$K_2' = \frac{K_2}{\sigma^2} e^{K_\sigma \tau} \tag{6.3.11}$$

where σ^2 is the expected power (this should not be confused with the measurement covariance), τ is the detector threshold (again in units of intensity), and

$$K_\sigma = \frac{1 - \sigma^2}{\sigma^2} \tag{6.3.12}$$

The variable transformation in this case is given by

$$t_{2a} = \ln(1 + K_2' e^{-\frac{1}{2} u_{2a}}) \tag{6.3.13}$$

where u_{2a} is the sum of random variables

$$u_{2a} = 2K_\sigma \varsigma + w_2 \tag{6.3.14}$$

The RV ς is a truncated exponential random variable defined by the PDF of (6.3.10); the random variable w_2, defined in (6.3.5), is just a χ_2^2 RV (i.e. another exponential). The final PDF that results is

$$p_{t_{2a}}(t_{2a}) = \begin{cases} \frac{1}{|K_\sigma|\sigma^2+1} e^{-|K_\sigma|\tau} \frac{\exp(t_{2a})}{K'} & t_{2a} < \ln(1 + K' e^{|K_\sigma|\tau}) \\ e^{\frac{\tau}{\sigma^2}} \frac{1}{|K_\sigma|\sigma^2+1} \left(\frac{K'}{\exp(t_{2a})-1} \right)^{\frac{1}{|K_\sigma|\sigma_2}} \frac{\exp(t_{2a})}{\exp(t_{2a})-1} & t_{2a} > \ln(1 + K' e^{|K_\sigma|\tau}) \end{cases} \tag{6.3.15}$$

This PDF is plotted in Figure 6.8. Again, repeated convolutions produce the PDF for the batch ML-PMHT LLR, and the result is shown in Figure 6.9. As compared to the 2-D case without amplitude (refer back to Figure 6.5), the addition of the

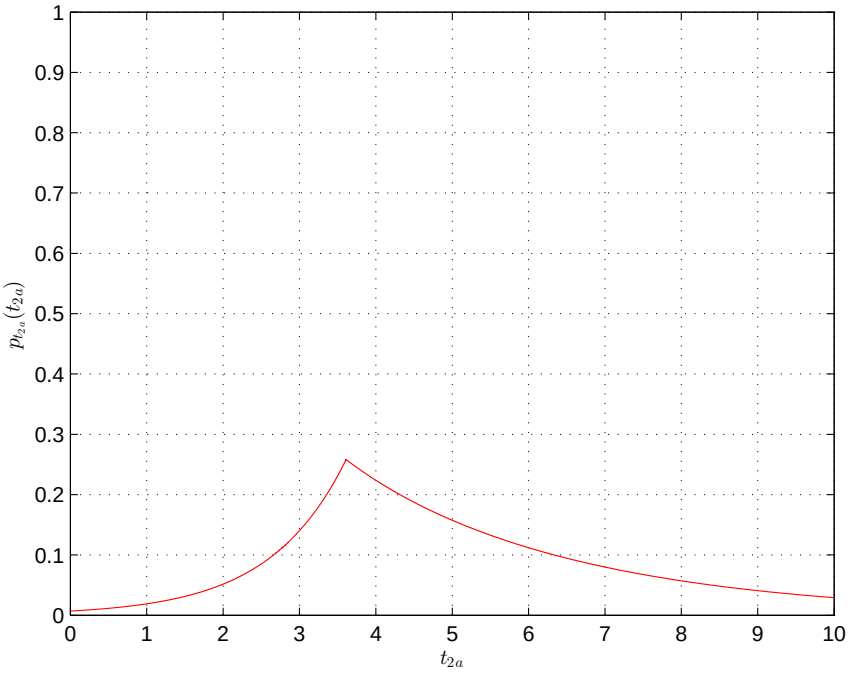

Fig. 6.8: Two-dimensional single-measurement PDF (target likelihood) with amplitude feature

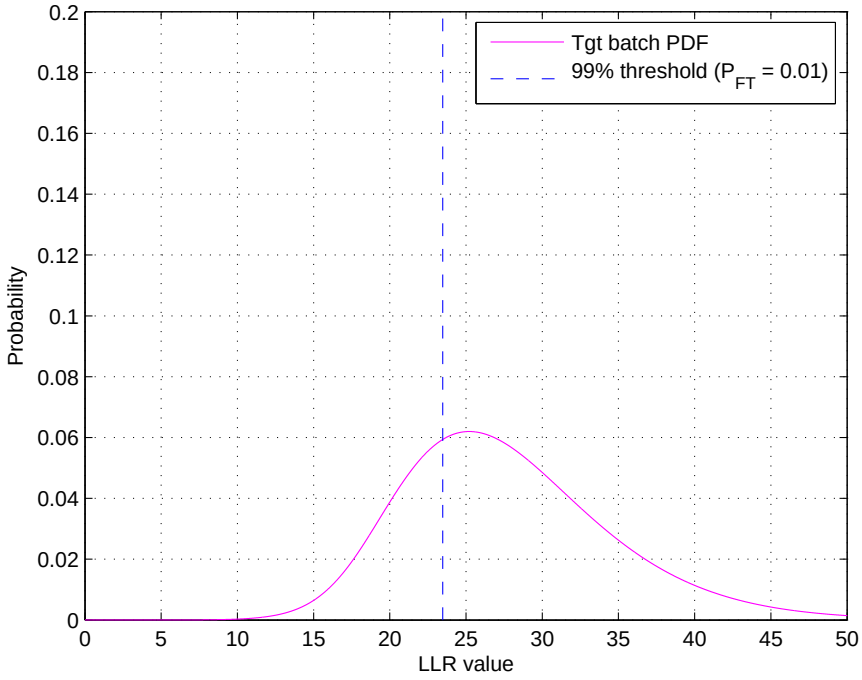

Fig. 6.9: Two-dimensional ML-PMHT target batch PDF (overall target likelihood) with amplitude feature. The threshold calculated from the peak clutter PDF is also plotted.

amplitude feature to the target has shifted the peak target PDF to the right as well as increased its variance. The effect on target trackability will be discussed in the next section.

6.3.3 Clutter and target peak PDFs as mixtures

All the of the work up to this point has been predicated on knowing N_t, the number of target measurements, and N_c, the number of clutter measurements. However, when making predictions, these values will not be known exactly; they can only be probabilistically described. Because of this, for both clutter and target, the peak PDFs will be created as mixture PDFs, made of individual peak PDFs for different values of N_t and N_c; the weights will be calculated from the probability assumptions for the target and clutter.

For the target, in a batch of N_W scans, P_d is the probability of target detection in any given scan. The expected number of target measurements then, is

$$E[N_t] = P_d N_W \tag{6.3.16}$$

The actual number of target measurements will have a binomial distribution with this value as the mean. These binomial probabilities become the weights for calculating the mixture PDF for the batch target PDF.

For clutter, the expected number of clutter measurements in a batch is

$$E[N_c] = N_W V \lambda \tag{6.3.17}$$

The actual number of clutter measurements will follow a Poisson distribution with this value as its mean. The Poisson PMF values become the weights for calculating a mixture PDF for the peak clutter likelihood.

Finally, with known values for N_c and N_t, it is possible to calculate probabilities π_0 and π_1. First, π_0 is given by

$$\pi_0 = \frac{N_c}{N_c + N_t} \tag{6.3.18}$$

and π_1 is given by

$$\pi_1 = \frac{N_t}{N_c + N_t} \tag{6.3.19}$$

6.4 Applications

Here, we apply the clutter PDFs developed in [61] along with the above work to develop TOC curves. These will then be used to analyze the effect of various parameters on trackability, such as clutter density, probability of detection, and measurement covariance.

For reference, the ML-PMHT parameters used[4] in this section are given in Table 6.1. These parameters were also used to create the plots in Section 6.2.

[4] In general, Table 6.1 gives parameters for all figures in this work that do *not* vary in a given plot. For instance, Figure 6.12 shows a 2-D plot of P_d vs. clutter density. Obviously, P_d is related to π_0 and π_1, so for this case, the table does not give the correct values for these parameters. However, all other parameters that do not vary in the plot, such as the search volumes and the measurement errors *are* given in the table.

Table 6.1: ML-PMHT parameters

1-Dimensional Parameters	
Angular volume	180°
Angular error	2°
Ave meas per scan	10
Number of scans	60
π_1	0.05
π_0	0.95
2-D and 3-D Parameters	
Angular volume	360°
Angular error σ_{az}	5°
Time delay volume	60 sec
Time delay error σ_t	0.1 sec
Range-rate volume	10 units per sec
Range-rate error $\sigma_{\tilde{r}}$	0.5 units per sec
Amplitude threshold	3 dB
Expected target amplitude	6 dB
Ave meas per scan	9.8
Number of scans	11
π_1	0.04
π_0	0.96

6.4.1 Tracker Operating Characteristic curves

We utilize the peak clutter PDFs developed in Chapter 5 and peak target PDFs to develop examples of a TOC curve. Such a curve is analogous to a receiver operating characteristic (ROC) curve from classical detection theory. Basically, the threshold is varied from low to high; for each threshold value, the exceedance probability is calculated by integrating the right tail (with the threshold as the lower integration limit) of the clutter and target peak PDFs. In this way, each threshold will result in a pair of values — the probability of target track detection (P_{DT}) and the probability of false track (P_{FT}) for the given threshold. The pairs are then plotted to form a TOC curve. Such curves can, for a given set of parameters, help determine if a target in a certain environment is "trackable."

An example of a TOC curve is presented in Figure 6.10. This is the result of 2-D processing (time-delay and azimuth); the parameters used to generate this plot are given in Table 6.1. Also included in the plot is the empirical TOC curve for identical ML-PMHT parameters. This was generated by simulating 5000 clutter-only samples and 5000 target-only samples, optimizing them with the ML-PMHT algorithm, and generating empirical peak clutter and peak target distributions. The empirical and model TOC curves match up well, indicating that the tracking model is accurate in predicting actual ML-PMHT tracker performance. (It should also be noted that it took approximately 20 hours to generate the empirical curve,

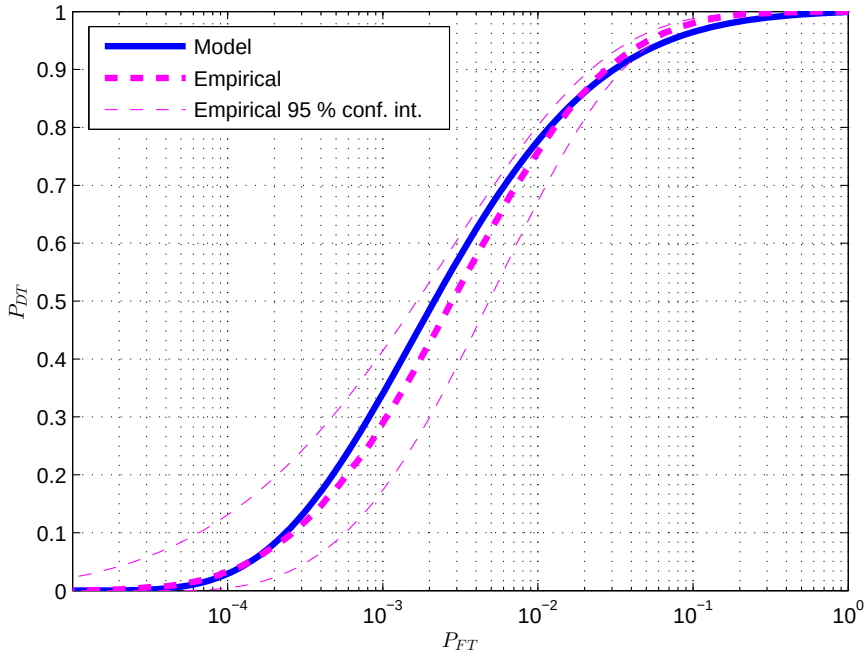

Fig. 6.10: Model and empirical Tracker Operating Characteristic curves for 2-D case

while it took only several seconds to produce the model curve. This ability to provide predictions in real-time is one of the big advantages of this work.)

6.4.2 Effect of batch length

We next generate a TOC curve and use it to calculate, for a certain set of parameters, the effect of batch length on performance. For a clutter density of $\lambda = 1 \times 10^{-8}$ clutter points per unit volume, a single target measurement detection probability of $P_d = 0.8$, and a threshold value of $P_{FT} = 0.01$, in-track

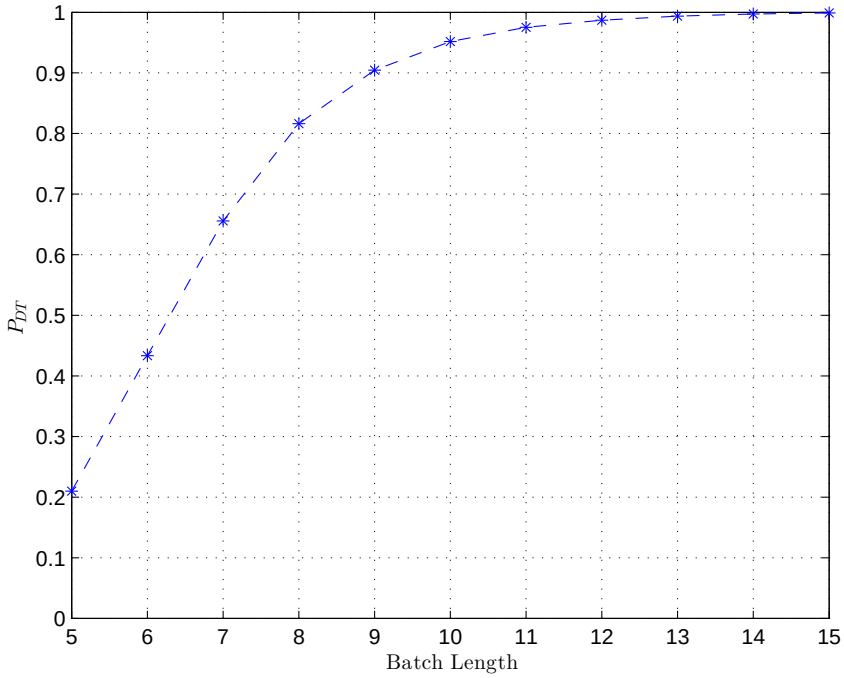

Fig. 6.11: In-track probability (P_{DT}) as a function of batch length.

probability (P_{DT}) values are calculated for a range of batch lengths. Results are shown in Figure 6.11. As the batch length increases, P_{DT} goes up. For this simple example, the model produces expected results.

6.4.3 Performance curves

At this point, the TOC curves are used to define a "trackability" metric. For a given (arbitrary) P_{FT} value, the corresponding P_{DT} value can be found from the TOC curve. In this work, the trackability metric is defined at $P_{FT} = 0.01$. A scenario is deemed to be trackable if the corresponding value of P_{DT} on the TOC

curve is greater than 0.5. In this manner, we can generate trackability curves that show, as a function of two parameters at a time, where the ML-PMHT algorithm is able to track and where it is not.

Using this methodology, a curve displaying trackable regions for spatial clutter density[5] (λ) versus the measurement detection probability, P_d, (note that this *not* P_{DT}) is presented in Figure 6.12. This plot shows that trackability in the 2-D case goes as expected: as P_d increases, the number of target-generated measurements in a batch increases, and this helps trackability. Likewise, as spatial clutter density increases, the number of clutter points increases, which will inhibit trackability unless P_d is high (i.e. a strong target). This is exactly what is seen.

Next, trackability is examined for the 3-D case (i.e. time-delay/azimuth/Doppler); the corresponding trackability plot is shown in Figure 6.13. It is instructive to compare the 3-D curve to the 2-D curve (which is overlaid on this plot). The ML-PMHT parameters are all the same for both curves (again, the constant parameters are those listed in Table 6.1). In the Doppler case, though, the trackability region is extended to the right which shows using Doppler is an advantage over all the tested values of P_d and λ (which is an expected result).

[5] Spatial clutter density is defined here as "clutter points per square distance unit" (distance units in this work are all arbitrary). For the plots showing Doppler results, λ remains in these units for purposes of comparison — we do not show the effect that the Doppler volume has on the clutter density on the plot labels, although this is (correctly) included in the model equations.

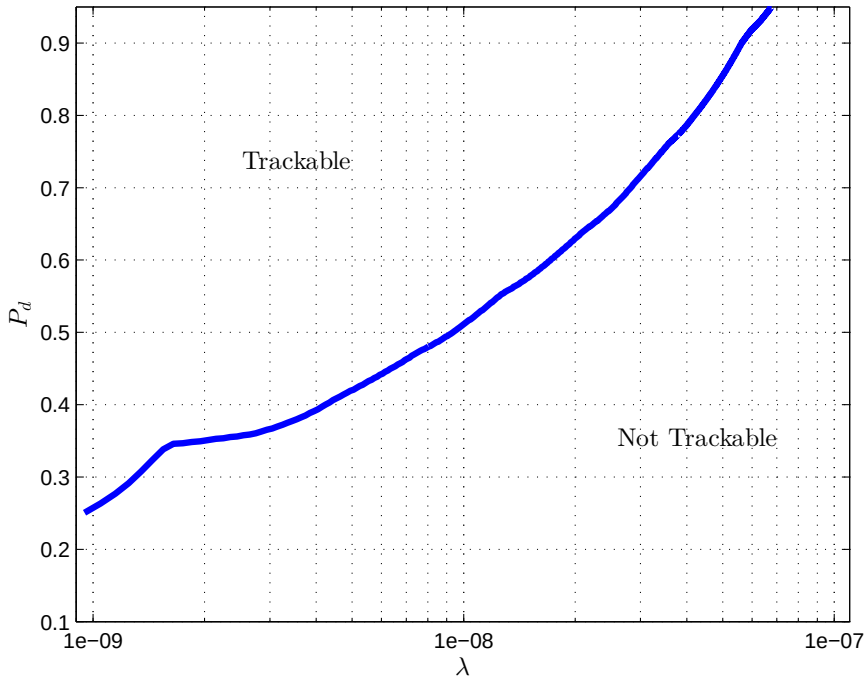

Fig. 6.12: Trackability plot for 2-D case

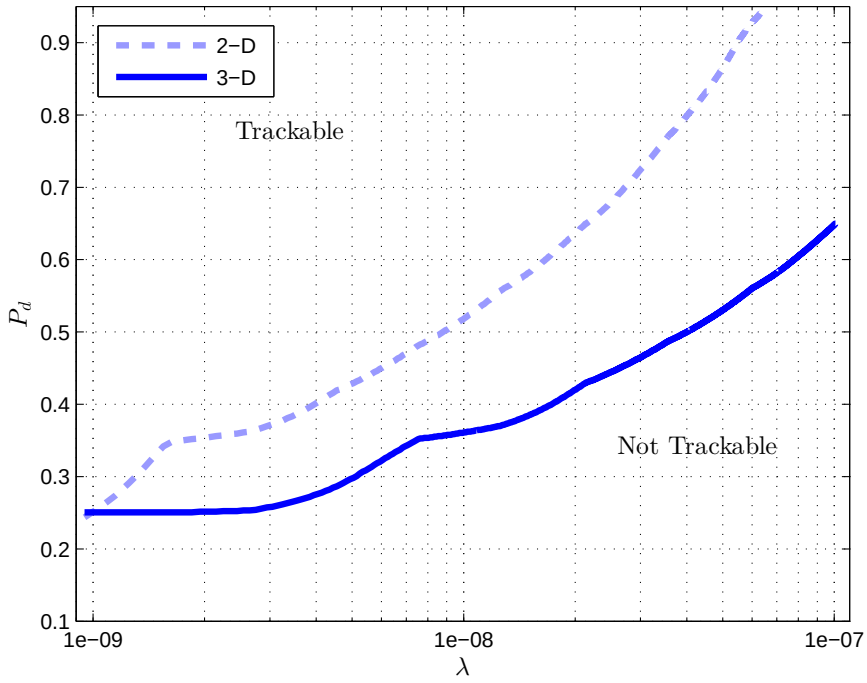

Fig. 6.13: Trackability plot for 3-D case, $\sigma_{Doppler} = 0.5$ units/sec. Curve for 2-D case is shown for comparison.

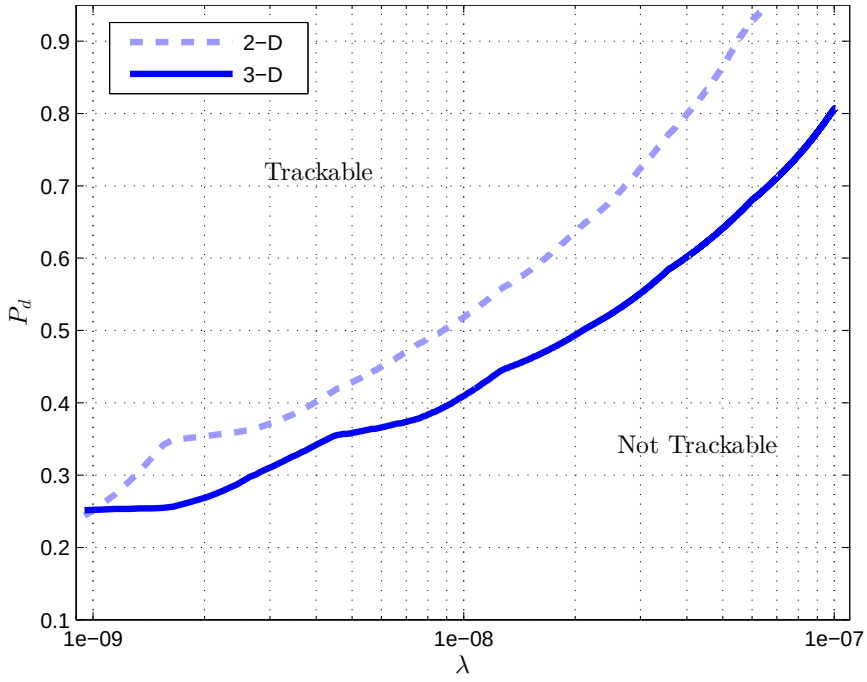

Fig. 6.14: Trackability plot for 3-D case, $\sigma_{Doppler} = 1.0$ units/sec. Curve for 2-D case is shown for comparison.

Now the Doppler measurement error is gradually increased to see if using Doppler always provides an advantage. Doppler measurement error is increased from 0.5 units/sec to 1 unit/sec to 2 units/sec to 5 units/sec; the corresponding trackability regions are shown in Figures 6.14, 6.15, and 6.16. In these figures, the 3-D trackability boundary slowly shifts to the left; as Doppler error increases, the trackability region decreases. Finally, note that the curve in Figure 6.16 showing results for $\sigma_{Doppler} = 5.0$ units/sec is actually *worse* than the 2-D case. In this circumstance, using Doppler actually degrades performance.

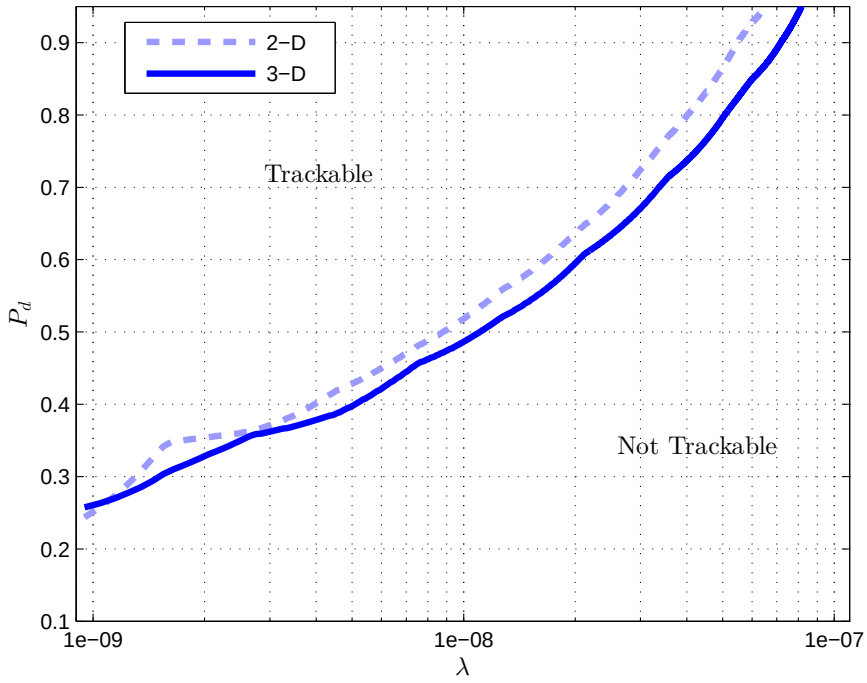

Fig. 6.15: Trackability plot for 3-D case, $\sigma_{Doppler} = 2.0$ units/sec. Curve for 2-D case is shown for comparison.

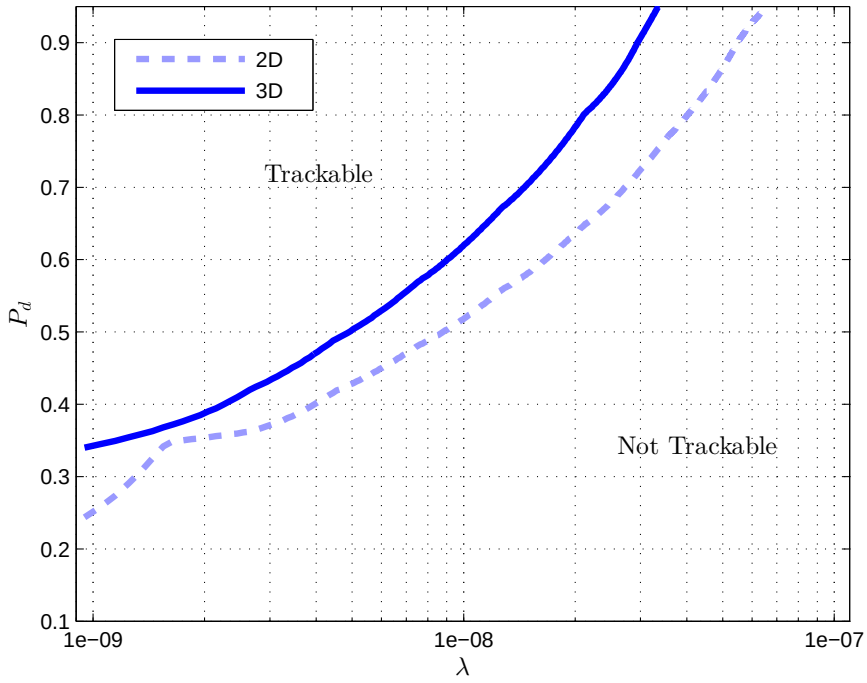

Fig. 6.16: Trackability plot for 3-D case, $\sigma_{Doppler} = 5.0$ units/sec. Curve for 2-D case is shown for comparison. Note in this case 2-D is *better* than 3-D.

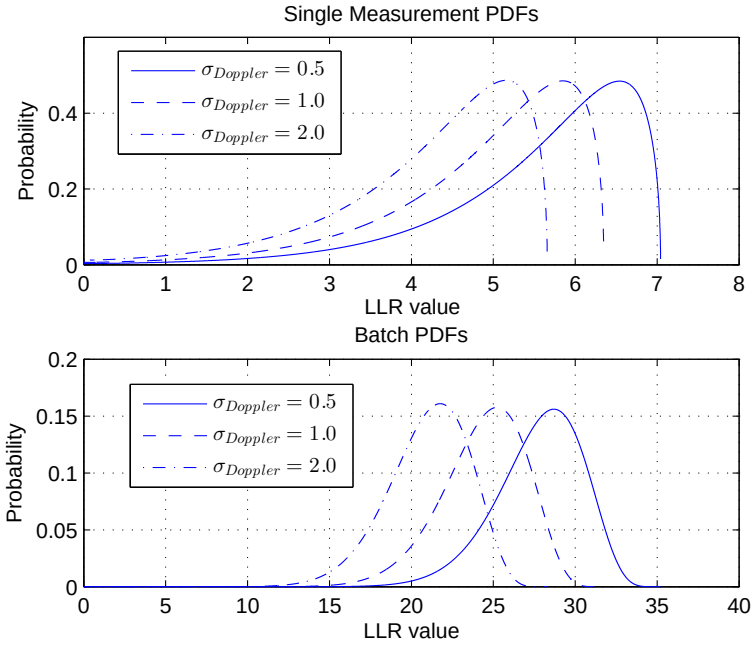

Fig. 6.17: Effect of changing $\sigma_{Doppler}$. Top plot shows single-measurement PDFs; bottom plot shows batch PDFs.

The mathematical reason for this can be seen in (6.3.9). The PDF $p_{t_3}(t_3)$ has an upper limit at $t_3 = \ln(1 + K_3)$. From (6.3.4), we can see that K_3 (in this case $d = 3$) is proportional to $V/\sigma_{Doppler}$. As $\sigma_{Doppler}$ gets bigger in relation to V, K_3 decreases, so the right-hand limit on the single-measurement PDF moves to the left. This is illustrated in the top plot of Figure 6.17. This shrinking of the single-measurement PDF causes the batch PDF to shift to the left as well, which is shown in the bottom plot of Figure 6.17. From an intuitive standpoint what is happening is that as the error increases in relation to the search volume, less and less useful information is available from the Doppler measurement. (A measurement ceases to be useful if its accuracy is so poor that all that can be said about it is that it is located somewhere in the measurement space.) Eventually the loss of information is such that it cannot make up for the added complexity (of processing Doppler) to the model; at this point, we are essentially overparameterizing the problem.

Next, the effect of varying the other measurement errors (azimuth and time) is examined. For two different P_d values, curves showing trackability as a function of σ_{az} versus λ are shown in Figure 6.18; curves showing trackability as a function of σ_t versus λ are shown in Figure 6.19.

For these cases, as the measurement errors get larger, less information is available from each measurement, so the trackability region should decrease. The effect of increasing λ is the same as above; a higher spatial clutter density will reduce

trackability. This is exactly what is seen in the resultant plots. Figure 6.19 seems to show the presence of a threshold effect as clutter density increases for both the $P_d = 0.5$ and $P_d = 0.8$ curves. However, testing revealed that if σ_t is made even smaller ($\sigma_t \approx 0.001$ sec), then the trackability curve will start to bend again to the right. It appears that there is no simple relationship between the ML-PMHT parameters and trackability (for instance, "below a σ_t of x a target cannot be tracked"). In general, the shapes of the peak clutter and target distributions are complicated functions of all the ML-PMHT parameters as well as the clutter Poisson and target binomial mixture PMFs; trackabililty must be assessed for each set of parameters independently.

The trackability analysis is completed by examining the effect of using amplitude information on tracking. This is done only for 2-D tracking — it is not possible to get closed-form expressions for 3-D tracking with amplitude. Two cases are examined. The first is for an expected target power of 7 dB and the second is for an expected target power of 10 dB (the detector threshold in both cases was set at $\tau = 5$ dB). Figure 6.20 shows these curves; they indicate that processing amplitude increases trackability by a rather substantial amount for ML-PMHT; the higher the expected target power, the better. However, some care must be taken when considering these amplitude results. They are predicated on knowing the expected target intensity σ^2 — in reality this value will never be known exactly.

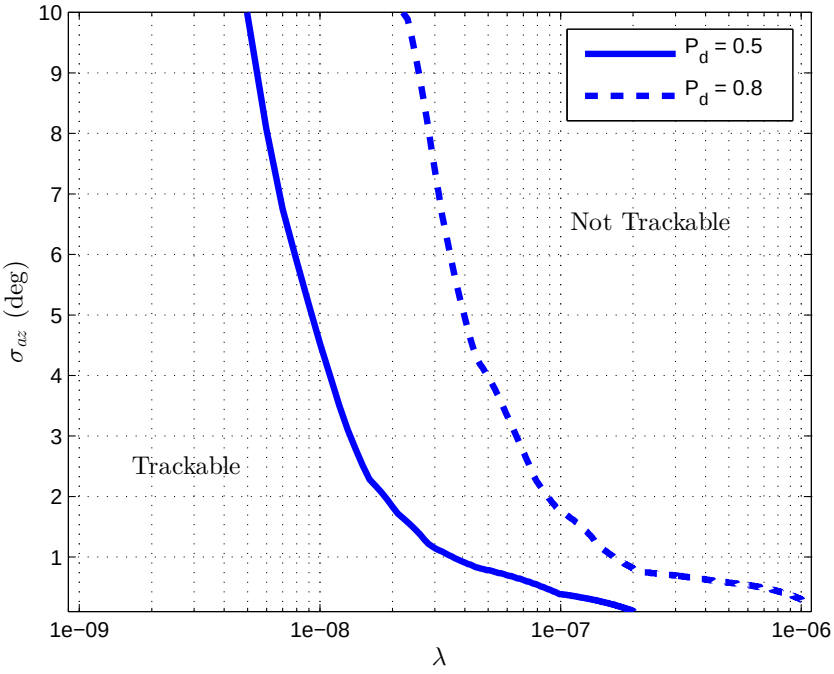

Fig. 6.18: Trackability plot for 2-D case as a function of σ_{az} and λ. Curves are shown for fixed P_d values of 0.5 and 0.8. Note that values of clutter density (λ) now go to higher value than in than previous plots.

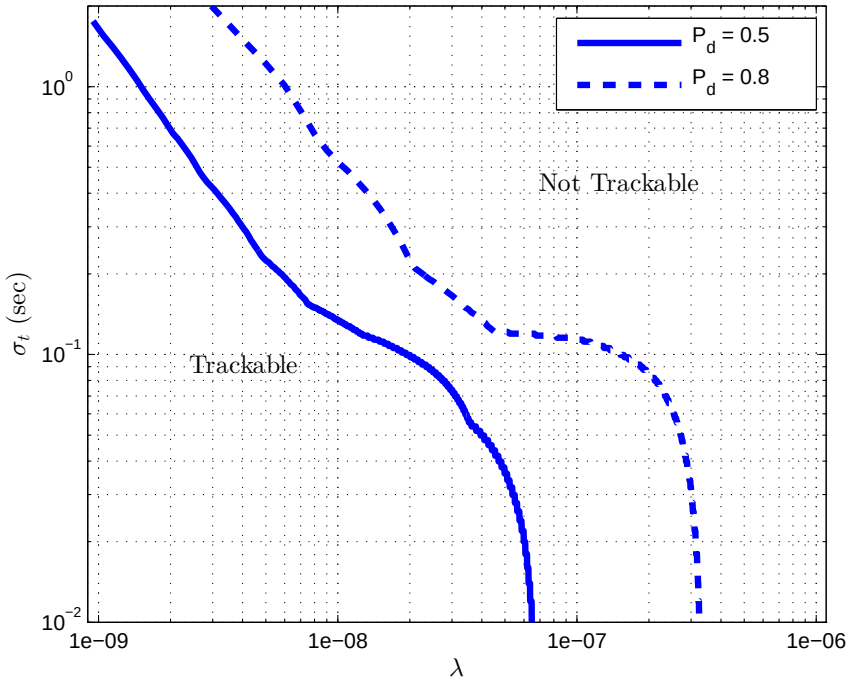

Fig. 6.19: Trackability plot for 2-D case as a function of σ_t and λ. Curves are shown for fixed P_d values of 0.5 and 0.8.

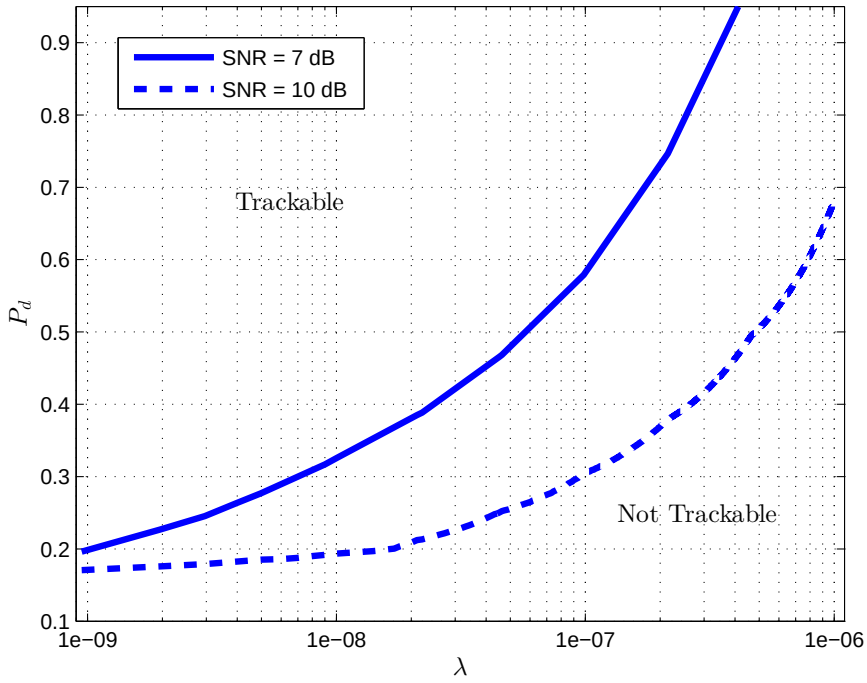

Fig. 6.20: Trackability plot for 2-D case with amplitude information. Curves are shown for expected target SNR values of 7 dB and 10 dB.

Additionally, this expected target intensity can be variable as well.

While beyond the scope of this paper, these observations point to some interesting follow-on work. First, the effect of not knowing the value of σ^2 should be examined in two cases; when the value is off by some constant amount, and when σ^2 changes (unpredictably) from measurement to measurement. (The latter case here is a realistic result of getting measurements in a batch that come from different source-receiver pairs.) Secondly, other types of clutter should be examined. The Rayleigh distribution has been used above (for amplitude) for both the clutter and the target. Recent work in [1], [2], and [3] has posited that the K-distribution better represents the amplitude from clutter returns in an underwater environment. The K-distribution is a much heavier-tailed PDF, which will make it more difficult to distinguish between the target and clutter with the amplitude feature. In contrast, the Rayleigh distribution is a very light-tailed distribution, which means that high-SNR measurements will be very heavily weighted towards the target. It is such weighting that causes the target PDF in Figure 6.9 to shift so far to the right (which is driving the increase in tracking performance). A heavier-tailed amplitude clutter distribution will cause this PDF to shift back to the left and close the gap between the peak clutter and target PDFs.

6.4.4 Other possible analysis

At this point, we have developed expressions for the peak points in the ML-PMHT LLR from both clutter and target (and applied them) for the following cases: 1-dimensional measurements, 2-dimensional measurements, 3-dimensional measurements, and 2-dimensional measurement with amplitude feature information, where the amplitude was assumed to have a Rayleigh distribution. These cases were presented because they all had a closed-form expression in the case of the ML-PMHT LLR "transforming" a single measurement, which provided for easy validation by alternate numerical methods. However, realistic cases for multistatic active tracking, primarily 3-dimensional measurements with amplitude information, or amplitude feature data that is not Rayleigh distributed, do not have closed-form expressions for their "single-measurement" PDFs. Nevertheless, such PDFs can be calculated numerically. At this point, the same methodology can be applied to find the peak distributions — either repeated convolutions or multiplication of the characteristic functions to get the PDF representing the ML-PMHT batch LLR for either clutter or a target. In the case of the target, the batch PDF is the peak PDF (its likelihood). In the case of clutter, extreme value theory can be used to develop the peak PDFs. These will give tracking thresholds, and the comparison of the peak target and clutter PDFs can be used to estimate target trackability. (In the previous section we discussed follow-on work in the

area of tracking with amplitude information — this type of numerical calculation will be needed in this case.)

It should be further stressed that this approach is *not* unique to ML-PMHT. There is no reason why this technique should not work for other maximum likelihood approaches. As long as the underlying distribution of the measurements is known or assumed, any likelihood ratio can be treated as just a random variable transformation, and the methods above can be used. While in many cases, the calculations will likely have to be done numerically rather than in closed form, there are no assumptions in the above approach that would preclude its more general application.

6.5 Conclusions

We have presented a novel method for determining the PDFs describing the maximum points in the ML-PMHT LLR caused by clutter (which was given in Chapter 5) as well as a target, using random variable transformations. The target peak PDF and the clutter peak PDF can be used to then create "tracker operating characteristic" or TOC curves and a trackability metric. This metric was used to first show performance as a function of P_d and λ for the 2-D case. Next, the 3-D case (i.e. including Doppler processing) was analyzed; it was shown that in general, processing Doppler improves performance, with the exception being when

the Doppler measurement error approaches the size of the Doppler measurement volume. After this, we gave examples of trackability performance as a function of azimuthal and time-delay measurement errors. Finally, we analyzed the effect that amplitude information has on tracking, and showed that in the case of Rayleigh-distributed clutter with known expected target intensity, using amplitude increases trackability. Overall, the ability to analyze a scenario for trackability under various conditions adds capability to the ML-PMHT multistatic active tracking framework.

Chapter 7

Concluding Thoughts

7.1 Overall conclusions

Over the last two decades, the ML-PDA tracker has been developed into a powerful multistatic active sonar tracker. In a sense, it is an "optimal" multistatic tracker in that can use all available data, and as long as the target and the environment are consistent with the algorithm's assumptions, no approximations need to be made to get it to function. The majority of this work focused on ML-PMHT, a slightly different tracker. The two algorithms are very similar; the important difference lies in their respective target assignment models. ML-PDA assumes that only zero or one measurement in a scan can come from a target, while ML-PMHT relaxes this and allows any number of measurements in a scan to come from the target. This has important implications, and it led to the work that improved the performance of (primarily) the ML-PMHT algorithm as a multistatic tracker.

To make such ML-PMHT improvements, we first created a true multitarget implementation of ML-PMHT. In the process, we developed an expression for the

ML-PMHT CRLB. We showed that ML-PMHT is a statistically efficient estimator, and that the ML-PMHT CRLB can be evaluated in real time, a valuable result in and of itself – this allows for a "complete" tracker framework that provides both a state estimate as well as a reliable estimate of the state covariance.

We then tested these developments by comparing ML-PDA and ML-PMHT with Monte Carlo testing. This testing first showed that ML-PDA and ML-PHMT are effective very low observable trackers, working down to an expected target SNR of 4-5 dB (post-signal processing). After this, the ML-PDA and the ML-PMHT tracking algorithms were applied to five different benchmark scenarios with Monte Carlo trials using a target measurement generation model of zero or one measurements originating from the target in a scan. For scenarios with a single target or multiple targets with measurements that could easily be differentiated by dynamics, the performances of ML-PDA and ML-PMHT were identical — ML-PMHT did not suffer from the fact that its measurement assignment model did not match the actual target measurement generation model. In cases with closely-spaced targets with measurements that could not be differentiated easily by dynamics, ML-PMHT outperformed ML-PDA due to the fact that the former had a true multitarget LLR formulation, while the latter had to handle multiple targets in a sequential single-target mode. Finally, when the target measurement generation model was switched to that of ML-PMHT, with multiple measurements

per scan being generated by the target, ML-PMHT outperformed ML-PDA in terms of the number of duplicate tracks generated. Overall, the performance of ML-PMHT makes it the preferred algorithm.

Next, we compared the performance of the ML-PMHT tracker in Cartesian measurement space and delay-bearing measurement space. (ML-PMHT and ML-PDA have an advantage over other trackers in that their LLRs can be implemented in either space.) As expected, the tracker performed far better in delay-bearing space, since in this space the measurement Gaussian distribution correctly represents the uncertainty. This shows that ML-PMHT (and any other trackers that can operate in delay-bearing space) have a significant advantage when operating on multistatic sonar data over trackers that work in Cartesian measurement space. In the low-observable target-amplitude regime of the scenario, ML-PMHT also benefited from being a batch tracker.

A maneuvering-model parameterization was then developed for ML-PMHT to increase the tracker's ability to follow rapidly-maneuvering targets. This parameterization, especially when implemented in a delay-bearing variable-batch length mode, significantly increased the tracker's performance in almost all metric areas. To go along with this, the expression previously developed for the straight-line ML-PMHT FIM/CRLB was extended for the maneuvering-model parameterization. Using this, the NEES calculated over Monte-Carlo trials showed that ML-PMHT

with the maneuvering-model parameterization is an efficient estimator. Again, this allows the use of the CRLB as a consistent estimate for the state covariance in an ML-PMHT tracking framework.

Lastly, we introduced a novel method for determining the PDFs describing the maximum points in the ML-PMHT LLR caused by clutter as well as a target (in the case of clutter, the use of extreme value theory is required). The EV distribution describing the peak point in the LLR due to clutter can be used to quickly and easily determine a tracking threshold — previously, this had to be done either through trial-and-error or with very time consuming optimizations of Monte Carlo simulations. Thresholds obtained in this manner match up extremely well with thresholds obtained by (the much more time consuming) Monte-Carlo simulation and optimization of ML-PMHT LLR surfaces. A similar methodology is used to determine the PDF of the peak point in the ML-PMHT LLR due to the target. The target EV PDF and the clutter EV PDF can be used to then create "tracker operating characteristic" curves that for known clutter and target parameters, will show whether or not a target is trackable. In general, these TOC curves can be used for parametric analysis of the ML-PMHT tracker. Overall, the ability to rapidly and accurately determine tracking thresholds as well as the ability to analyze a scenario for trackability under various conditions add capability to the ML-PMHT multistatic active tracking framework.

7.2 Future work

The results of this dissertation point the way to an important area of future work. The issue of trackability that was explored in Section 6.4.1 can be used to explore further the effects of using amplitude feature information in the tracker. Such future work was discussed in detail in Section 6.3.2. To summarize, the effects of amplitude on tracking should be examined when amplitude likelihoods follow a Rayleigh distribution but the expected amplitude is not exactly known, and when the clutter amplitude likelihood does not even follow a Rayleigh distribution, but rather, a more heavily-tailed PDF, such as a K-distribution. Such conditions match up more realistically with actual multistatic tracking conditions and will give an indication of how useful the amplitude feature really is for this type of tracking.

A second area of future exploration involves determining the effect of the "contact lens" problem [56], [60] on trackability. This problem occurs for trackers that receive range-bearing (or similar) measurements yet operate in Cartesian space. The measurement errors are approximated as Gaussian in Cartesian space; however, at long ranges this becomes an inaccurate approximation. The "true" measurement uncertainty region (in Cartesian space) at long ranges looks more like a banana in two dimensions or a contact lens in three dimensions. The work of [60] examined this effect empirically using Monte-Carlo testing. The work in Chapters 5 and 6

will be used to model the effect that this contact lens problem has on trackability.

7.3 Final thoughts

Working on the ML-PDA and the ML-PMHT trackers over the past five years has been quite an experience. We hope that we have added in some meaningful way to the overall body of work in the area of multistatic multitarget tracking. We would like to think that this dissertation does not represent the conclusion of work on ML-PDA and ML-PMHT, but rather, just milestone in a process of continuous exploration.

Appendix A

Calculating the ML-PMHT Fisher Information Matrix

Here we fully derive the Fisher Information Matrix (FIM) for a window of ML-PMHT data. This is a specific implementation of the general work given in [57]. The derivation is also similar to that done for ML-PDA in [41]. However, the ML-PMHT calculation ends up being at most a 4-fold integral (as opposed to approximately an m-fold integral for ML-PDA, where m is the number of measurements in a scan), so this can be done real-time instead of having to calculate it offline with Monte-Carlo integration as must be done with ML-PDA.

Consider a window of ML-PMHT data with N_w scans. Since all measurements from scan to scan are assumed to be independent, the FIM $\mathbf{J}$ of the total window can just be written as the sum of the FIM's $\mathbf{J}_i$ of the individual scans,

$$\mathbf{J} = \sum_{i=1}^{N_w} \mathbf{J}_i \tag{A.1}$$

Now, the FIM of a scan is calculated as

$$\mathbf{J}_i = E\{[\nabla_x \ln p[Z(i)|\mathbf{x}]][\nabla_x \ln p[Z(i)|\mathbf{x}]]^T\}|_{\mathbf{x}=\mathbf{x}_{true}} \tag{A.2}$$

The ML-PMHT likelihood for a single scan is

$$p[Z(i)|\mathbf{x}] = \prod_{j=1}^{m_i} \left\{ \frac{\pi_0}{V} p_0^\tau(a_j) + \pi_1 p[\mathbf{z}_j(i)|\mathbf{x}] p_1^\tau(a_j) \right\} \tag{A.3}$$

Taking the gradient of the logarithm of this gives

$$\nabla_x \ln p[Z(i)|\mathbf{x}] = \sum_{j=1}^{m_i} \mathbf{D}_\phi^T \frac{\frac{\pi_1 p_1^\tau(a_j)}{\sqrt{|2\pi\mathbf{R}_j|}} e^{-\frac{1}{2}(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)} \mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)}{\frac{\pi_0 p_0^\tau(a_j)}{V} + \frac{\pi_1 p_1^\tau(a_j)}{\sqrt{|2\pi\mathbf{R}_j|}} e^{-\frac{1}{2}(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)}} \tag{A.4}$$

Here, ϕ is the state-to-measurement conversion and $\mathbf{D}_\phi$ is the Jacobian of ϕ. (Note that we have dropped the scan index i on the values $\mathbf{z}$, ϕ, $\mathbf{R}$ and a for ease of expression.) Inserting (A.4) into (A.2) and recognizing that cross terms in the product of the sums will go to zero results in

$$\mathbf{J}_i = \sum_{j=1}^{m_i} \int_V \mathbf{D}_\phi^T \frac{\frac{[\pi_1 p_1^\tau(a_j)]^2}{|2\pi\mathbf{R}_j|} e^{-(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)} \mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}}{p(\mathbf{z}_j|\mathbf{x})^2} \times \cdots$$
$$\mathbf{D}_\phi \prod_{l=1}^{m_i} p(\mathbf{z}_l|\mathbf{x}) d\mathbf{Z} d\mathbf{a} \tag{A.5}$$

where $d\mathbf{Z} = \prod_{l=1}^{m_i} d\mathbf{z}_l$ and $d\mathbf{a} = \prod_{l=1}^{m_i} da_l$. We are denoting the denominator in (A.4) as $p(\mathbf{z}_j|\mathbf{x})$. Note that all $p(\mathbf{z}_l|\mathbf{x})$ terms for $l \neq j$ are separable and simply integrate to one, and the term for $l = j$ is canceled by one of the terms in the denominator. This happens because the likelihood function for ML-PMHT (for a scan) is a product, and this reduces the overall FIM integral from being an m-fold integral to only a 3- or 4-fold integral. In contrast, as is shown in [41], the ML-PDA likelihood function for a scan of data consists of a summation, not a product, over all the measurements in the scan. As a result, taking the gradient

of the logarithm of this results in summation over all the measurements in the denominator of the ML-PDA equivalent of (A.5). Thus, the cancellation that occurs above does not happen for ML-PDA; its FIM integral remains on the order of m-fold.

At this point, we can reduce (A.5) to

$$\mathbf{J}_i = \sum_{j=1}^{m_i} \int_V \mathbf{D}_\phi^T \frac{\frac{[\pi_1 p_1^\tau(a_j)]^2}{|2\pi\mathbf{R}_j|} e^{-(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)} \mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}}{\frac{\pi_0 p_0^\tau(a_j)}{V} + \frac{\pi_1 p_1^\tau(a_j)}{\sqrt{|2\pi\mathbf{R}_j|}} e^{-\frac{1}{2}(\mathbf{z}_j-\phi)^T\mathbf{R}_j^{-1}(\mathbf{z}_j-\phi)}} \mathbf{D}_\phi d\mathbf{z}_j da_j \tag{A.6}$$

The Jacobians are independent of the variables of integration and the summation indices, and thus can be moved outside of the integral and summation. Furthermore, we can make following substitution of

$$\xi_j = \mathbf{G}_j(\mathbf{z}_j - \phi) \tag{A.7}$$

where $\mathbf{G}_j$ is the Cholesky decomposition of $\mathbf{R}_j^{-1}$ such that $\mathbf{G}_j^T\mathbf{G}_j = \mathbf{R}_j^{-1}$. With this, it is possible to write the final expression for $\mathbf{J}_i$:

$$\mathbf{J}_i = \mathbf{D}_\phi^T \sum_{j=1}^{m_i} \mathbf{G}_j^T \int_V \frac{\frac{[\pi_1 p_1^\tau(a_j)]^2}{|2\pi\mathbf{R}_j|} e^{-\xi_j^T\xi_j} \xi_j\xi_j^T}{\frac{\pi_0 p_0^\tau(a_j)}{V} + \frac{\pi_1 p_1^\tau(a_j)}{\sqrt{|2\pi\mathbf{R}_j|}} e^{-\frac{1}{2}\xi_j^T\xi_j}} d\xi_j da_j \frac{\mathbf{G}_j}{|\mathbf{G}_j|} \mathbf{D}_\phi \tag{A.8}$$

This is a relatively simple 3- or 4-fold integration (depending on the dimension of $\mathbf{z}_j$) in this application and can be done in real time. The resultant $\mathbf{J}_i$ for each scan is simply summed up then in accordance with (A.1) to get the FIM for a window of data.

Appendix B

Calculating Maneuver-Model Jacobian Entries

Here, we present the components for the entries of the Jacobian given in (4.4.5). For Cartesian measurement space, the first entry in measurement space (Cartesian x-position) is

$$\phi_1 = x_0 + T_{00}\dot{x} + S_{01}\dot{y} \tag{B.1}$$

where $T_{ij} = t_m + (t - t_m)R_{ij}$ and $S_{ij} = (t - t_m)R_{ij}$. (Here, R_{ij} is the $(i,j)^{th}$ entry of the rotation matrix with the maneuver angle as the rotation angle.) The derivatives with respect to the positions and velocities are

$$\frac{\partial \phi_1}{\partial x_0} = 1 \qquad \frac{\partial \phi_1}{\partial \dot{x}} = T_{00} \qquad \frac{\partial \phi_1}{\partial y_0} = 0 \qquad \frac{\partial \phi_1}{\partial \dot{y}} = S_{01} \tag{B.2}$$

The derivatives with respect to the maneuver time and angle are

$$\frac{\partial \phi_1}{\partial t_m} = \Delta\dot{x} \quad \frac{\partial \phi_1}{\partial \theta_m} = -\Delta t \dot{y}_{pm} \tag{B.3}$$

where $\Delta\dot{x} = \dot{x} - \dot{x}_{pm}$ (i.e. the difference in x-velocity pre- and post-maneuver), and $\Delta t = t - t_m$. The next component in the predicted measurement (Cartesian

y-position) is

$$\phi_2 = S_{01}\dot{x} + y_0 + T_{11}\dot{y} \tag{B.4}$$

For this, the derivatives with respect to position and velocity are

$$\frac{\partial\phi_2}{\partial x_0} = 0 \qquad \frac{\partial\phi_2}{\partial\dot{x}} = S_{01} \qquad \frac{\partial\phi_2}{\partial y_0} = 1 \qquad \frac{\partial\phi_2}{\partial\dot{y}} = T_{11} \tag{B.5}$$

The maneuver time and angle derivatives are

$$\frac{\partial\phi_2}{\partial t_m} = \Delta\dot{y} \qquad \frac{\partial\phi_2}{\partial\theta_m} = \Delta t\dot{x}_{pm} \tag{B.6}$$

Finally, ϕ_3, the bistatic range-rate, is given by [26], and its derivative components all take the form

$$\frac{\partial\phi_3}{\partial a} = \frac{1}{2}\left(\frac{\partial\phi_3^S}{\partial a} + \frac{\partial\phi_3^R}{\partial a}\right) \tag{B.7}$$

Here, $a \in \{x_o, \dot{x}, y_0, \dot{y}, t_m, \theta_m\}$, and the S and R superscripts signify the source and receiver components of the expression. Letting $\tilde{r}$ represent the "generic" ϕ_3 (i.e. a common expression that can be used for both the source and the receiver components)

$$\frac{\partial\tilde{r}}{\partial x_0} = \frac{y_{pm}^2\dot{x}_{pm} - x_{pm}y_{pm}\dot{y}_{pm}}{r_{pm}^3} \tag{B.8}$$

Here, once again, r is the range from the target to the source or receiver, and the pm subscripts signify "post-maneuver." Furthermore, it is understood that the positions and velocities have the source or receiver positions and velocities subtracted out (depending on the component for which the equation is being

used). The remaining derivatives are

$$\frac{\partial \tilde{r}}{\partial \dot{x}} = \frac{R_{00}(x_{pm}^3 + y_{pm}^2 x_{pm}) + R_{10}(x_{pm}^2 y_{pm} + y_{pm}^3)}{r_{pm}^3} + \cdots \\ \frac{T_{00}(y_{pm}^2 \dot{x}_{pm} - x_{pm} y_{pm} \dot{y}_{pm}) + S_{10}(x_{pm}^2 \dot{y}_{pm} - x_{pm} y_{pm} \dot{x}_{pm})}{r_{pm}^3} \tag{B.9}$$

$$\frac{\partial \tilde{r}}{\partial y_0} = \frac{x_{pm}^2 \dot{y}_{pm} - x_{pm} y_{pm} \dot{x}_{pm}}{r_{pm}^3} \tag{B.10}$$

$$\frac{\partial \tilde{r}}{\partial \dot{y}} = \frac{R_{01}(x_{pm}^3 + y_{pm}^2 x_{pm}) + R_{11}(x_{pm}^2 y_{pm} + y_{pm}^3)}{r_{pm}^3} + \cdots \\ \frac{T_{11}(x_{pm}^2 \dot{y}_{pm}^2 - x_{pm} y_{pm} \dot{x}_{pm}) + S_{01}(y_{pm}^2 \dot{x}_{pm} - x_{pm} y_{pm} \dot{y}_{pm})}{r_{pm}^3} \tag{B.11}$$

$$\frac{\partial \tilde{r}}{\partial t_m} = \frac{(x_{pm}\Delta\dot{y} - y_{pm}\Delta\dot{x})(x_{pm}\dot{y}_{pm} - y_{pm}\dot{x}_{pm})}{r_{pm}^3} \tag{B.12}$$

$$\frac{\partial \tilde{r}}{\partial \theta_m} = \frac{[x_{pm}^2 + y_{pm}^2 - \Delta t(x_{pm}\dot{x}_{pm} + y_{pm}\dot{y}_{pm})][y_{pm}\dot{x}_{pm} - x_{pm}\dot{y}_{pm}]}{r_{pm}^3} \tag{B.13}$$

Expressions for the delay-bearing implementation can be written as functions of the above Cartesian expressions. Introducing the superscripts C and DB to differentiate between the Cartesian and delay-bearing versions, we have,

$$\phi_1^{DB} = \tan^{-1}\left(\frac{\Delta\phi_2^C}{\Delta\phi_1^C}\right) \tag{B.14}$$

where

$$\Delta\phi_1^C = \phi_1^C - x_R \qquad \Delta\phi_2^C = \phi_2^C - y_R \tag{B.15}$$

Here, x_R and y_R are again the Cartesian positions of the receiver. We can write a generic formula for the Jacobian entries in delay-bearing form that is a function

of the respective derivatives from the Cartesian form

$$\frac{\partial \phi_1^{DB}}{\partial a} = \frac{\Delta\phi_1^C \frac{\partial \phi_2^C}{\partial a} - \Delta\phi_2^C \frac{\partial \phi_1^C}{\partial a}}{r_{TR}^2} \tag{B.16}$$

The predicted delay-time measurement is

$$\phi_2^{DB} = \frac{r_{TS} + r_{TR}}{c} \tag{B.17}$$

where c is the speed of sound. Again, we can write a generic formula for the delay-bearing form of the Jacobian entries from the Cartesian terms

$$\frac{\partial \phi_2^{DB}}{\partial a} = \left[\frac{(x - x_S)\frac{\partial \phi_1^C}{\partial a} + (y - y_S)\frac{\partial \phi_2^C}{\partial a}}{r_{TS}} + \frac{(x - x_R)\frac{\partial \phi_1^C}{\partial a} + (y - y_R)\frac{\partial \phi_2^C}{\partial a}}{r_{TR}} \right] \tag{B.18}$$

The Jacobian entries for ϕ_3 (range-rate) are identical for the Cartesian and delay-bearing cases.

Finally, the above expressions can also be used for any pre-maneuver measurements with the following adjustments: $T_{i,j} \rightarrow t$, $S_{i,j} \rightarrow 0$, and $\Delta t \rightarrow 0$. All derivatives with respect to maneuver time and maneuver angle are set to zero.

Appendix C

Calculating the Moments of $\sqrt{K}$

Here, we derive approximations for the expected value and the variance of $\sqrt{K}$, where K is a Poisson random variable with parameter ϕ. The variance that results is rather interesting — it is constant, independent of ϕ. The work in this appendix is similar to that used in deriving the Anscombe transform [5], [34].

Start with the PMF for K, which is written as

$$P\{K = k\} = \frac{\phi^k e^{-\phi}}{k!} \tag{C.1}$$

Now, we want to calculate the mean and variance of $\sqrt{K}$. The calculation of the mean is

$$E[\sqrt{K}] = \sum_{k=0}^{\infty} \sqrt{k} \frac{\phi^k e^{-\phi}}{k!} \tag{C.2}$$

This sum cannot be simplified (exactly) to a closed-form expression; however it can be seen by inspection that the sum converges (each term is less than the corresponding term in the infinite summation shown below in (C.13) for $E[k]$, which converges to ϕ). We break the $k-$dependent terms in (C.2) into two terms

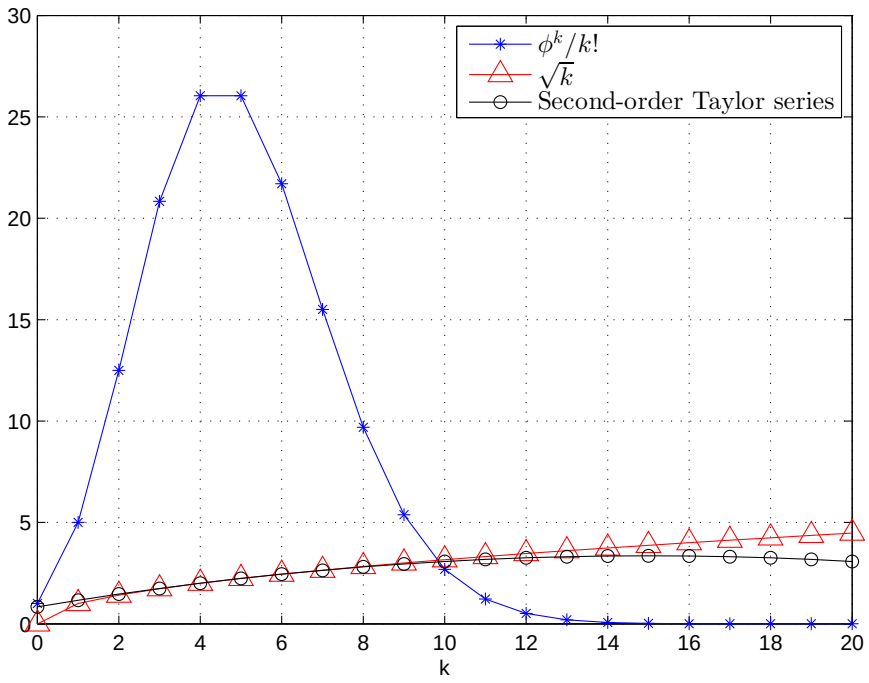

Fig. C.1: Components of the summation for $E[\sqrt{K}]$

$$\sum_{k=0}^{\infty} \sqrt{k} \frac{\phi^k e^{-\phi}}{k!} = e^{-\phi} \sum_{k=0}^{\infty} f_1(k) f_2(k) \tag{C.3}$$

where

$$f_1(k) = \sqrt{k} \tag{C.4}$$

and

$$f_2(k) = \frac{\phi^k}{k!} \tag{C.5}$$

These two components are shown on Figure C.1. At this point, a second-order Taylor series is developed for $f_1(k)$ around the mode of $f_2(k)$. The value of the mode is determined by calculating the location of the maximum for $f_2(k)$. This is done by using Stirling's approximation

$$k! \approx \frac{1}{\sqrt{2\pi k}} k^k e^{-k} \tag{C.6}$$

Inserting (C.6) into (C.5) yields

$$f_2(k) \approx \frac{1}{\sqrt{2\pi k}} \left(\frac{\phi e}{k} \right)^k \tag{C.7}$$

The maximum is found by taking the derivative with respect to k of the logarithm of this expression and setting it to zero (note that this is treating $f(k)$ as being continuous). After some algebra, this results in

$$\phi = k e^{\frac{1}{2k}} \tag{C.8}$$

It is reasonable to assume that $1/2k$ is small, so we can make the further approximation that

$$e^{\frac{1}{2k}} \approx \left(1 + \frac{1}{2k} \right) \tag{C.9}$$

With this, it is possible to solve for the location of the mode, denoted as k_0

$$k_0 = \phi - \frac{1}{2} \tag{C.10}$$

At this point, $f_1(k)$ will be approximated with a second order Taylor series around $k_0 = \phi - \frac{1}{2}$. Such a Taylor series is given as

$$f_1(k) \approx f_1(k_o) + f_1'(k_0)(k - k_0) + \frac{1}{2} f_1''(k_0)(k - k_0)^2 \tag{C.11}$$

Taking the first and second derivatives of $f_1(k)$ and working through some algebra gives

$$f_1(k) = \frac{3}{8}\left(\phi - \frac{1}{2}\right)^{\frac{1}{2}} + \frac{3}{4}\frac{k}{(\phi - \frac{1}{2})^{\frac{1}{2}}} - \frac{1}{8}\frac{k^2}{(\phi - \frac{1}{2})^{\frac{3}{2}}} \tag{C.12}$$

This approximation is shown in Figure C.1 — it can be seen that for values of k where the product of $f_1(k)$ and $f_2(k)$ is non-zero, the approximation is a good fit to $\sqrt{k}$.

Now, the first two moments of K (not $\sqrt{K}$) are written out as

$$E[K] = \sum_{k=0}^{\infty} k \frac{\phi^k}{k!} e^{-\phi} = \phi \tag{C.13}$$

and

$$E[K^2] = \sum_{k=0}^{\infty} k^2 \frac{\phi^k}{k!} e^{-\phi} = \phi^2 + \phi \tag{C.14}$$

Combining (C.3), (C.5), (C.12), (C.13), and (C.14) gives

$$\sum_{k=0}^{\infty} \sqrt{k} \frac{\phi^k}{k!} e^{-\phi} \approx \frac{3}{8}\left(\phi - \frac{1}{2}\right)^{\frac{1}{2}} + \frac{3}{4}\frac{\phi}{(\phi - \frac{1}{2})^{\frac{1}{2}}} - \frac{1}{8}\frac{\phi^2 + \phi}{(\phi - \frac{1}{2})^{\frac{3}{2}}} \tag{C.15}$$

After some algebra, this expression becomes

$$\sum_{k=0}^{\infty} \sqrt{x} \frac{\phi^k}{k!} e^{-\phi} \approx \frac{\phi - \frac{3}{8}}{(\phi - \frac{1}{2})^{\frac{1}{2}}} - \frac{3}{32} \frac{1}{(\phi - \frac{1}{2})^{\frac{3}{2}}} \tag{C.16}$$

The second term on the right-hand side of this equation is small compared to the first term and is discarded, leaving

$$\sum_{k=0}^{\infty} \sqrt{x} \frac{\phi^k}{k!} e^{-\phi} \approx \frac{\phi - \frac{3}{8}}{(\phi - \frac{1}{2})^{\frac{1}{2}}} \tag{C.17}$$

From here, performing polynomial division and more algebra gives

$$\sum_{k=0}^{\infty} \sqrt{x} \frac{\phi^k}{k!} e^{-\phi} \approx \sqrt{\phi - \frac{1}{4} + \frac{1}{64} \frac{1}{(\phi - \frac{1}{2})}} \tag{C.18}$$

Again, the second term of the radicand is small and is neglected. This leads to the final value for the expectation of $\sqrt{K}$

$$E[\sqrt{K}] = \sum_{k=0}^{\infty} \sqrt{x} \frac{\phi^k}{k!} e^{-\phi} = \sqrt{\phi - \frac{1}{4}} \tag{C.19}$$

With this, the variance of $\sqrt{K}$ is easily calculated. Starting with the definition

$$\text{var}(\sqrt{K}) = E[\sqrt{K}^2] - E^2[\sqrt{K}] \tag{C.20}$$

The first term on the right is recognized as just the expected value of the original Poisson RV, so this gives

$$\text{var}(\sqrt{K}) \approx \phi - \left(\sqrt{\phi - \frac{1}{4}} \right)^2 \tag{C.21}$$

This leaves a (surprisingly) simple final result

$$\text{var}(\sqrt{K}) \approx \frac{1}{4} \tag{C.22}$$

This expectation and variance of $\sqrt{K}$ are then used as necessary.

Bibliography

[1] D. Abraham. Detection-threshold approximation for non-Gaussian backgrounds. *IEEE Journal of Oceanic Engineering*, 35(2):355–365, 2010.

[2] D. Abraham and A. Lyons. Novel physical interpretations of K-distributed reverberation. *IEEE Journal of Oceanic Engineering*, 27(4):800–813, 2002.

[3] D. Abraham and A. Lyons. Simulation of non-Rayleigh reverberation and clutter. *IEEE Journal of Oceanic Engineering*, 29(2):347–362, 2004.

[4] A. Abramowitz and I. Stegun. Handbook of mathematical functions. In *National Bureau of Standards, Applied Math. Series*, 55, chapter 9.1.1, 9.1.98, and 9.12. Dover Publications, 1965.

[5] F. Anscombe. The transformation of Poisson, binomial and negative-binomial data. *Biometrika*, 35(3-4):246–254, 1948.

[6] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[7] D. Avitzour. A maximum likelihood approach to data association. *IEEE Transactions on Aerospace and Electronic Systems*, 28(2):560–566, 1996.

[8] Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association.* Academic Press, Orlando, FL, 1988.

[9] Y. Bar-Shalom and X. Li. *Multitarget-Multisensor Tracking: Principles and Techniques.* YBS Publishing, 1995.

[10] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation.* John Wiley and Sons, Inc., New York, NY, 2001.

[11] Y. Bar-Shalom, P. Willett, and X. Tian. *Tracking and Data Fusion: A Handbook of Algorithms*. YBS Publishing, 2011.

[12] S. Benen and P. Berkel. Fusion concepts for multistatic sonar systems. In *Workshop on Sensor Data Fusion: Trends, Solutions, Applications Workshop*, Berlin, 2011.

[13] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 2008.

[14] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems*. Artech House, Boston, MA, 1999.

[15] W. Blanding, P. Willett, and Y. Bar-Shalom. Covert sonar tracking. In *Proceedings of the IEEE Aerospace Conference*, Big Sky, March 2005.

[16] W. Blanding, P. Willett, and Y. Bar-Shalom. UUV-platform cooperation in covert tracking. In *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets, #5913-1U*, San Diego, April 2005.

[17] W. Blanding, P. Willett, and Y. Bar-Shalom. Multiple target tracking using maximum likelihood probabilistic data association. In *Proceedings of the IEEE Aerospace Conference*, Big Sky, March 2007.

[18] W. Blanding, P. Willett, and Y. Bar-Shalom. Offline and real-time methods for ML-PDA track validation. *IEEE Transactions on Signal Processing*, 55(5):1994–2006, 2007.

[19] W. Blanding, P. Willett, and Y. Bar-Shalom. ML-PDA: Advances and a new multitarget approach. *EURASIP Journal on Advances in Signal Processing*, 2008:1–13, 2008.

[20] W. Blanding, P. Willett, and S. Coraluppi. Sequential ML for multistatic sonar tracking. In *Proceedings Oceans 2007 Europe Conference*, Aberdeen, Scotland, June 2007.

[21] C.G. Broyden. The convergence of a class of double-rank minimization algorithms. *Journal of the Institute of Mathematics and its Applications*, 9:76–90, 1970.

[22] E. Castillo. *Extreme Value Theory in Engineering*. Harcourt Brace Jovanovich, Boston, 1988.

[23] M. R. Chummun, Y. Bar-Shalom, and T. Kirubarajan. Adaptive early-detection ML-PDA estimator for LO targets with EO sensors. *IEEE Transactions on Aerospace and Electronic Systems*, 38(2):694–707, 2002.

[24] S. Coles. *An Introduction to Statistical Modeling of Extreme Values*. Springer-Verlag, London, 2001.

[25] S. Coraluppi. Mulitstatic sonar localization. *IEEE Journal of Oceanic Engineering*, 31(4):964–974, 2006.

[26] H. Cox. Fundamentals of bistatic active sonar. Technical Report W1038, BBN Systems and Technologies Corporation, 1988.

[27] M. Daun and F. Ehlers. Tracking algorithms for multistatic sonar systems. *EURASIP Journal on Advances in Signal Processing*, 2010(1):1–28, 2010.

[28] H. David. *Order Statistics*. John Wiley & Sons, Inc., New York, 1981.

[29] A. Demster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, 39(1):1–38, 1977.

[30] P. Embrechts, C. Klüppelberg, and T. Mikosch. *Modelling Extremal Events for Insurance and Finance*. Springer-Verlag, Berlin, 1997.

[31] O. Erdinc, P. Willett, and Y. Bar-Shalom. The bin-occupancy filter and its connection to the PHD filters. *IEEE Transactions on Signal Processing*, 57(11):4232–4246, 2006.

[32] R. Fletcher. A new approach to variable metric algorithms. *Computer Journal*, 13(3):317–322, 1970.

[33] T. Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE Journal of Oceanic Engineering*, 8(3):173–184, 1983.

[34] F. Freeman and J. Tukey. Transformations related to the angular and the square root. *The Annals of Mathematical Statistics*, 21(4):607–611, 1950.

[35] R. Georgescu, P. Willett, and S. Schoenecker. GM-CPHD and MLPDA applied to the SEABAR07 and TNO-blind multi-static sonar data. In *Proceedings of the 12th International Conference on Information Fusion*, Seattle, WA, 2009.

[36] R. Georgescu, P. Willett, and S. Schoenecker. GM-CPHD and ML-PDA applied to the Metron multi-static sonar dataset. In *Proceedings of the 13th International Conference on Information Fusion*, Edinburgh, Scotland, 2010.

[37] D. Goldfarb. A family of variable metric update derived by variational means. *Mathematics of Computation*, 24:23–26, 1970.

[38] E. Gumbel. *Statistics of Extremes*. Columbia Univerisity Press, New York, 1958.

[39] C. Jauffret and Y. Bar-Shalom. Track formation with bearing and frequency measurements in clutter. In *Proceedings of the 29th Conference on Decision and Control*, Honolulu, December 1990.

[40] C. Jauffret and Y. Bar-Shalom. Track formation with bearing and frequency measurements in clutter. *IEEE Transactions on Aerospace and Electronic Systems*, 26(6):999–1010, 1990.

[41] T. Kirubarajan and Y. Bar-Shalom. Low observable target motion analysis using amplitude information. *IEEE Transactions on Aerospace and Electronic Systems*, 32(4):1637–1382, 1996.

[42] T. Kurien. Issues in the design of practical multitarget tracking algorithms. In Y. Bar-Shalom, editor, *Multitarget-Multisensor Tracking: Advanced Applications*. Artech House, Norwood, MA, 1990.

[43] M. Leadbetter, G. Lindgren, and H. Rootzén. *Extremes and Related Properties of Random Sequences and Processes*. Springer Verlag, New York, 1983.

[44] R. Mahler. Multitarget Bayes filtering via first-order mulitarget moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152–1178, 2003.

[45] R. Mahler. PHD filters of higher order in target number. *IEEE Transactions on Aerospace and Electronic Systems*, 43(4):1523–1543, 2007.

[46] J. Marcum. A statistical theory of target detection by pulsed radar. Technical report, The Rand Corporation, Santa Monica, CA, March 1947. Research Memorandum RM-745.

[47] J. Marcum. A statistical theory of target detection by pulsed radar: Mathematical appendix. Technical report, The Rand Corporation, Santa Monica, CA, March 1948. Research Memorandum RM-753.

[48] N. Mukhopadhyay. *Probability and Statistical Inference*. Marcel Dekker, New York, 2000.

[49] J. Neyman and E. Pearson. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London*, Series A, Containing Papers of a Mathematical of Physical Character(232):289–337, 1933.

[50] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, New York, 2000.

[51] K. Orlov. Description of the Metron simulation data set for MSTWG. 2009.

[52] A. Poore and A. Robertson III. A new Lagrangian relaxation based algorithm for a class of multidimensional assignment problems. *Computational Optimization and Applications*, 8, 1997.

[53] A. Poore, S. Lu, and B. Suchomel. Data association using multiple frame assignments. In *Handbook of Multisensor Data Fusion*. CRC Press, LLC, 2001.

[54] M. Powell. Restart procedures of the conjugate gradient method. *Mathematical Programming*, 12:241–254, 1977.

[55] D. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, AC-24:843–854, 1979.

[56] K. Romeo, P. Willett, and Y. Bar-Shalom. Particle filter tracking for long range radars. In *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets, #8393-OF*, Baltimore, MD, 2012.

[57] Y. Ruan, P. Willett, and R. Streit. A comparison of the PMHT and PDAF tracking algorithms based on their model CRLBs. In *Proceedings of the SPIE Aerospace Conference on Acquisition, Tracking and Pointing XIII, #3528-60*, Orlando, FL, 1999.

[58] S. Schoenecker, P. Willett, and Y. Bar-Shalom. Maximum likelihood probabilistic data association tracker applied to bistatic sonar data sets. In *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets, #7698-19*, Orlando, FL, 2010.

[59] S. Schoenecker, P. Willett, and Y. Bar-Shalom. A comparison of the ML-PDA and the ML-PMHT algorithms. In *Proceedings of the 14th International Conference on Information Fusion*, Chicago, IL, 2011.

[60] S. Schoenecker, P. Willett, and Y. Bar-Shalom. Maximum likelihood probabilistic multi-hypothesis tracker applied to multistatic sonar data sets. In *Proceedings of the SPIE Conference on Signal Processing, Sensor Fusion, and Target Recognition, #8050-9*, Orlando, FL, 2011.

[61] S. Schoenecker, P. Willett, and Y. Bar-Shalom. Extreme-value analysis for ML-PMHT, part 1: Threshold determination for false track probability. *Submitted to IEEE Transactions on Aerospace and Electronic Systems*, 2013.

[62] S. Schoenecker, P. Willett, and Y. Bar-Shalom. Extreme-value analysis for ML-PMHT, part 2: Probability of target track detection and the fundamental trackability of targets. *Submitted to IEEE Transactions on Aerospace and Electronic Systems*, 2013.

[63] S. Schoenecker, P. Willett, and Y. Bar-Shalom. ML-PDA and ML-PMHT: Comparing multistatic sonar trackers for VLO targets using a new multitarget implementation. *To appear the Journal of Oceanic Engineering*, 2013.

[64] D. Shanno. Conditioning of Quasi-Newton methods for function minimization. *Mathematical Computation*, 24:647–656, 1970.

[65] D. Shanno. Conjugate gradient methods with inexact searches. *Mathematics of Operations Research*, 3:244–256, 1978.

[66] R. Streit. Multisensor multitarget intensity filter. In *Proceedings of the 11th International Conference on Information Fusion*, Cologne, Germany, 2008.

[67] R. Streit. *Poisson Point Processes*. Springer, 2010.

[68] R. Streit and T. Luginbuhl. A probabilistic multi-hypothesis tracking algorithm without enumeration. In *Proceedings of the 6th Joint Data Fusion Symposium*, Laurel, MD, June 1993.

[69] R. Streit and T. Luginbuhl. Maximum likelihood method for probabilistic multi-hypothesis tracking. In *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets, #2235*, Orlando, FL, 1994.

[70] R. Streit and T. Luginbuhl. Probabilistic multi-hypothesis tracking. Technical Report TR 10428, Naval Undersea Warfare Center, 1995.

[71] P. Swerling. Probability of detection for fluctuating targets. Technical report, The Rand Corporation, Santa Monica, CA, March 1954. Research Memorandum RM-1217.

[72] X. Tian and Y. Bar-Shalom. Robust tracking for very long range radars, part I: Algorithm comparisons. In *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets, #6699-0G*, San Diego, August 2007.

[73] X. Tian, Y. Bar-Shalom, G. Chen, E. Blasch, and K. Pham. A novel filtering approach for the general contact lens problem with range rate measurements. In *Proceedings of the 12th International Conference on Information Fusion*, Edinburgh, July 2010.

[74] K. Wilkens and M. Daun. A Gaussian mixture motion model and contact fusion applied to the Metron data set. In *Proceedings of the 14th International Conference on Information Fusion*, Chicago, IL, 2011.

[75] P. Willett. University of Connecticut Department of Electrical & Computer Engineering website, August 2012.

[76] P. Willett and S. Coraluppi. Application of the MLPDA to bistatic sonar. In *Proceedings of the IEEE Aerospace Conference*, Big Sky, March 2005.

[77] P. Willett and S. Coraluppi. MLPDA and MLPMHT applied to some MSTWG data. In *Proceedings of the 9th International Conference on Information Fusion*, Florence, Italy, July 2006.

[78] P. Willett, S. Coraluppi, and W. Blanding. Comparison of soft and hard assignment ML trackers on multistatic data. In *Proceedings of the IEEE Aerospace Conference*, Big Sky, March 2006.